Migration von SSIS ETL-Jobs zu AWS

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Migration von SSIS ETL-Jobs zu AWS

Table of Contents

Migration von SSIS ETL-Jobs zu AWS

Durga Prasad und Harpreet Singh, Amazon Web Services ()AWS

Dezember 2021 (Geschichte der Dokumente)

Amazon Web Services (AWS) bietet Services für die Entwicklung und Ausführung von Extraktions-, Transformations- und Ladeaufträgen (ETL) oder Extrahieren, Laden und Transformieren (ELT) und Big-Data-Workloads. Microsoft SQL Server Integration Services (SSIS) ist ein lokales ETL-Tool mit einer grafischen Oberfläche zum Erstellen von ETL-Jobs. Abhängig von der Architektur des Zielstatus können Sie die AWS Glue grafische Oberfläche oder das Apache Spark-Framework mit Python-Bibliotheken verwenden, um ETL-Jobs in der AWS Cloud zu erstellen.

In diesem Handbuch werden die Schritte für die Migration von SSIS-ETL/ELT-Jobs von lokal zu beschrieben. AWS Es werden Migrationsphasen, bewährte Verfahren und Empfehlungen zur Reduzierung des Migrationsaufwands und zur Verbesserung der Benutzererfahrung erörtert. Die Informationen gelten für jede AWS Zielarchitektur.

Der Leitfaden richtet sich an Programm- oder Projektmanager, Produkteigentümer, Lösungsarchitekten und Entwickler, die:

- Die Migration von SSIS zu erfolgt aus verschiedenen AWS Gründen, z. B. zur Modernisierung oder zur Kosteneinsparung.
- In erster Linie werden AWS Dienste für ETL- und Big-Data-Workloads verwendet.
- Sie suchen nach einer cloudbasierten Lösung, um von einem serverlosen Modell und einer flexiblen Preisgestaltung zu profitieren.

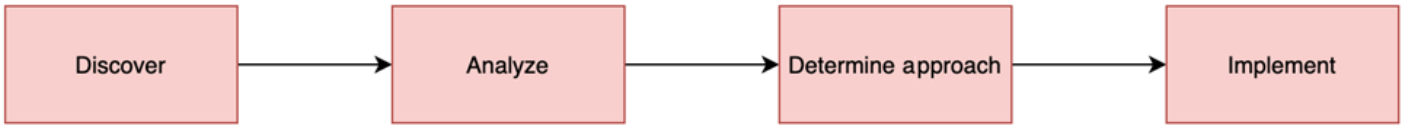
Gezielte Geschäftsergebnisse

Durch die Migration Ihrer SSIS-Jobs in die AWS Cloud können Sie die folgenden wichtigsten Ergebnisse erzielen:

- Modernisieren Sie bestehende ETL-Prozesse vor Ort auf kostengünstige Weise
- Reagieren Sie schneller auf die Informationsbedürfnisse des Unternehmens
- Führen Sie bestehende Prozesse zusammen und entfernen Sie ungenutzte Prozesse, um den Wartungs- und Betriebsaufwand zu reduzieren
- Schaffen Sie eine Grundlage, um die future Datenanforderungen Ihres Unternehmens zu erfüllen

Migrationsphasen

Die Migration von SSIS-ETL-Prozessen in die AWS Cloud besteht aus vier Hauptphasen: Erkennung, Analyse, Festlegung des Ansatzes und Implementierung.



Diese Phasen werden in den folgenden Abschnitten behandelt:

- Entdeckungsphase
- Phase der Analyse
- Festlegung des Migrationsansatzes
- Umsetzungsphase

Entdeckungsphase

In der Ermittlungsphase erstellen Sie eine Liste der SSIS-Pakete, zu denen Sie migrieren möchten. AWS Verschiedene Entwicklungsteams folgen bei der Entwicklung von ETL-Jobs unterschiedlichen Stilen, Standards und Mustern. Wir empfehlen Ihnen, die vorhandenen Dokumente Ihres Unternehmens zu lesen, um diese Muster zu verstehen. Die Dokumentation ist jedoch häufig unvollständig. Sie können die Extraktion wichtiger Informationen aus den ETL-Skripten automatisieren. Dies spart manuellen Aufwand und Zeit, reduziert menschliche Fehler und standardisiert den Migrationsansatz. Hier sind einige der wichtigen Details, die Sie extrahieren sollten:

- Gesamtzahl der Kontrollflussaufgaben
- Einzelheiten der Kontrollflussaufgaben
- Gesamtzahl der Datenflussaufgaben
- Verwendete Datenflusstransformationen
- Event-Handler
- Verbindungsmanager

Verwenden Sie diese Informationen, um die in Ihrem Unternehmen verwendeten ETL-Muster zu verstehen, ihre Komplexität zu bewerten und den geeigneten AWS Service für die Migration dieser Informationen zu ermitteln.

Die Migration dieser ETL-Details aus SSIS macht den Großteil des Migrationsaufwands aus. Zusätzliche Eigenschaften können jedoch Einblicke in Design- und Architekturentscheidungen bieten. Einige dieser SSIS-Eigenschaften sind:

- Prüfpunkte, die in SSIS verwendet werden, um Jobs nach einem Ausfall neu zu starten
- Propagieren Sie Variablen, die dazu beitragen, dass ein SSIS-Paket in bestimmten Anwendungsfällen erfolgreich ist, auch wenn ein Fehler auftritt
- Isolationsstufen für Transaktionen, mit denen die Qualität der aus Datenbanken gelesenen Daten gesteuert wird
- Protokollierung, um zu verstehen, welche Arten von Protokollen im aktuellen Design erfasst werden und an welchen Speicherorten

Das Ergebnis der Ermittlungsphase kann eine Bestandsaufnahme sein, wie die folgende Tabelle zeigt.

inventory

count	Package	Flow	Task
1	SSIS_Package_1	control_flow	Microsoft.ExecuteSQLTask
1	SSIS_Package_1	data_flow	Microsoft.BDD
1	SSIS_Package_1	data_flow	Microsoft.ConditionalSplit
2	SSIS_Package_1	data_flow	Microsoft.OLEDBDestination
1	SSIS_Package_1	data_flow	Microsoft.OLEDBSource
1	SSIS_Package_2	control_flow	Microsoft.DbMaintenanceFileCleanupTask
1	SSIS_Package_2	control_flow	Microsoft.ExecuteSQLTask
1	SSIS_Package_2	control_flow	Microsoft.ScriptTask
1	SSIS_Package_2	control_flow	STOCK:FOREACHLOOP
1	SSIS_Package_2	control_flow	STOCK:FORLOOP

Dieses Inventar kann die folgenden Informationen enthalten:

- Package: Name des zu migrierenden SSIS-Pakets
- Fluss: Kontrollfluss oder Datenfluss
- Aufgabe: Name der Ablaufsteuerungsaufgabe oder Datenflusskomponente
- Anzahl: Häufigkeit, mit der eine Aufgabe im SSIS-Paket verwendet wurde

Analysephase

Durch die Analyse der Informationen aus der Ermittlungsphase können Sie die Muster besser verstehen und die Zielarchitektur besser entwerfen. Folgen Sie diesen Hinweisen, um das Analyseergebnis zu verbessern:

- Die Protokollierungsoption auf Paketebene protokolliert Ereignisse und erfasst Laufzeitinformationen für jede Komponente. Verwenden Sie die benutzerdefinierte Protokollierung, damit Sie Protokolldateien so konfigurieren können, dass sie zusätzliche Informationen wie Ausführungs-IDs und andere Laufzeitwerte enthalten.
- Leiten Sie fehlerhafte Datensätze zur Analyse, Korrektur und Wiederverarbeitung an einen anderen Speicherort um.
- Machen Sie sich mit den Strategien zur Datenarchivierung und -löschung vertraut, die in Ihrer lokalen SSIS-Umgebung implementiert sind.
- Senden Sie Warnungen und Benachrichtigungen mithilfe von Aufgaben, die in SSIS enthalten sind (z. B. der Task E-Mail senden), oder mithilfe benutzerdefinierter Skripts (wie dem Skripttask).
- Machen Sie sich mit dem Verhalten von Transformationen im Gültigkeitsbereich vertraut, um beschädigte Daten zu vermeiden. Bei der Transformation für Lookup handelt es sich beispielsweise um eine Gleichverknüpfung zwischen zwei Datenmengen, und beim Vergleich wird zwischen Groß- und Kleinschreibung unterschieden.

Sie können jeden Eintrag im Inventar einer geschätzten Komplexität zuordnen, entweder in Bezug auf Dauer (Minuten oder Stunden) oder Stufe (einfach, mittel oder komplex). Dieses erweiterte Inventar, das in der folgenden Tabelle dargestellt ist, ist das Ergebnis der Analysephase.

inventory

count	Package	Flow	Task	Effort	Complexity
1	SSIS_Package_1	control_flow	Microsoft.ExecuteSQLTask	30	S
1	SSIS_Package_1	data_flow	Microsoft.BDD	0	N/A
1	SSIS_Package_1	data_flow	Microsoft.ConditionalSplit	30	S
2	SSIS_Package_1	data_flow	Microsoft.OLEDBDestination	60	M
1	SSIS_Package_1	data_flow	Microsoft.OLEDBSource	30	S
1	SSIS_Package_2	control_flow	Microsoft.DbMaintenanceFileCleanupTask	60	M
1	SSIS_Package_2	control_flow	Microsoft.ExecuteSQLTask	30	S
1	SSIS_Package_2	control_flow	Microsoft.ScriptTask	120	C
1	SSIS_Package_2	control_flow	STOCK:FOREACHLOOP	30	S
1	SSIS_Package_2	control_flow	STOCK:FORLOOP	30	S

Festlegung des Migrationsansatzes

Um sich für einen Migrationsansatz zu entscheiden, verwenden Sie die Analyse, die Sie in der vorherigen Phase anhand vorhandener Muster durchgeführt haben. Die future Daten- und Analyseanforderungen Ihres Unternehmens sind ebenso wichtige Überlegungen. Herkömmliche lokale ETL-Tools befassen sich mit relationalen Datenmodellen und strukturierten Daten. Wenn Sie halbstrukturierte und unstrukturierte Daten verarbeiten müssen, können Sie AWS Services wie AWS Glue Amazon EMR für die Migration verwenden. Zu den weiteren Faktoren, die den Migrationsansatz beeinflussen können, gehören:

- Egal, ob Sie eine grafische Oberfläche (wie AWS Glue Studio) oder ein benutzerdefiniertes Framework (wie Spark/Python-Bibliotheken) verwenden möchten
- Ob Sie sicheren Zugriff auf lokale Quellen und Ziele haben AWS
- Fähigkeiten und Schulungen, die für das Team erforderlich sind
- Audit- und Compliance-Anforderungen

Sie können zwischen drei Migrationsansätzen wählen: Big Bang, Phased und Lift and Shift. In der folgenden Tabelle werden diese drei Ansätze verglichen.

Ansatz	Beschreibung	Anwendungsfall	Vor- und Nachteile
Urknall	Migrieren Sie alle SSIS-Pakete innerhalb eines bestimmten Zeitraums .	• Komplexität, Umfang und Zielarchitektur liegen auf der Hand. • Das Team verfügt über die erforderl ichen Fähigkeiten oder die Lernkurve ist flach.	• Hohes Risiko. • Nimmt weniger Zeit in Anspruch als der schrittweise Ansatz. • Sie können Amazon EMR oder benutzerd efinierte Framework s verwenden AWS Glue.
Stufenweise	Identifizieren Sie ein SSIS-Paket für jedes unterschiedliche Muster und jede Komplexität. Migrieren Sie das Paket in die bestehende Architekt ur AWS, testen Sie es und vergleichen Sie die Ergebnisse mit dieser.	• Zeit ist kein Hindernis. • Sie möchten unterschiedliche Designs für unterschiedliche ETL-Muster.	• Weniger riskant als der Urknall-Ansatz, erfordert aber mehr Zeit und Mühe. • Sie können Amazon EMR oder benutzerd efinierte Framework s verwenden AWS Glue.
Heben und verschieb en	Migrieren Sie die aktuelle Architektur unverändert AWS.	• Ihre lokale Hardware wird nicht mehr unterstützt. • Sie verfügen nicht über die Ressourcen, um eine Migration sofort zu planen.	• Geringster Migrationsaufwand und geringster Zeitaufwand. • Die Probleme mit der bestehenden Lösung bestehen weiterhin AWS. • SSIS-Pakete werden unverände rt ausgeführt. Es

Ansatz	Beschreibung	Anwendungsfall	Vor- und Nachteile
			werden keine anderen ETL-Tools oder Frameworks benötigt.

Ein Vergleich der Daten auf den Quell- und Zielsystemen ist für eine erfolgreiche Migration von grundlegender Bedeutung. Da das bestehende Produktionssystem regelmäßig von den Quellsystemen aktualisiert wird, kann dieser Vergleich verwirrend sein. Aus diesem Grund empfehlen wir, dass Sie sich bei der Festlegung Ihres Migrationsansatzes auch für Ihre Datenvalidierungsstrategie entscheiden.

- Erstellen Sie zu einem bestimmten Datum und zu einer bestimmten Uhrzeit Backups aller zutreffenden Datenbanken und Dateien aus der Produktionsumgebung auf dem Quellsystem.
- Erstellen Sie Backups aller Datenbanken aus der Produktionsumgebung auf dem Zielsystem, nachdem alle Jobs erfolgreich Daten aus den gesicherten Quelldaten geladen haben.
- Stellen Sie die Quelldaten in einer Testumgebung wieder her und führen Sie die neuen Jobs aus.
- Vereinbaren Sie einen Prozentsatz gültiger Unterschiede zwischen den Quell- und Zieldatenbanken (alte und neue). Sie könnten beispielsweise entscheiden, dass ein Unterschied von weniger als 1% akzeptabel ist.
- Führen Sie alle Validierungsregeln auf, die abgedeckt werden sollen.
- Automatisieren Sie den Vergleich so weit wie möglich und decken Sie alle Regeln ab.

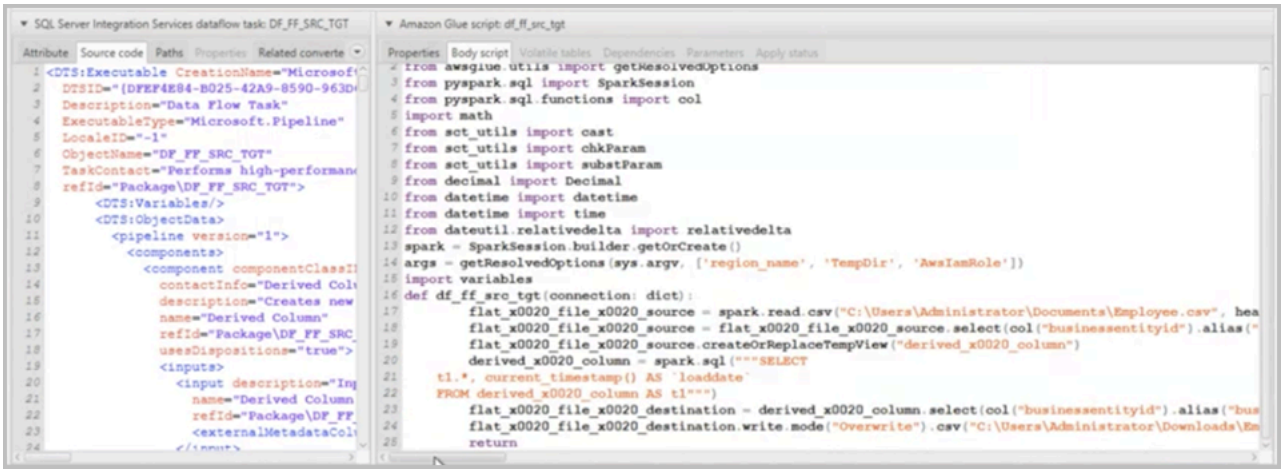
Umsetzungsphase

Eine Migration, die dem Urknall- oder Phasenansatz folgt, erfordert neue Entwicklungen und Tests. Das AWS Schema Conversion Tool (AWS SCT) kann automatisch AWS Glue Jobs aus SSIS-Paketen generieren. Dies reduziert den Zeit- und Arbeitsaufwand für die Migration erheblich. Oder Sie können AWS Glue Studio für die Entwicklung auf grafischer Oberfläche verwenden oder Spark-Bibliotheken erstellen, die Sie entweder auf Amazon EMR AWS Glue oder auf Amazon EMR ausführen können.

Die folgenden Abschnitte enthalten nützliche Hinweise zur Verwendung von AWS SCT AWS Glue, und Amazon EMR.

AWS SCT

Die folgende Bildschirmdarstellung zeigt ein AWS Glue Jobskript, das von konvertiert wurde. AWS SCT

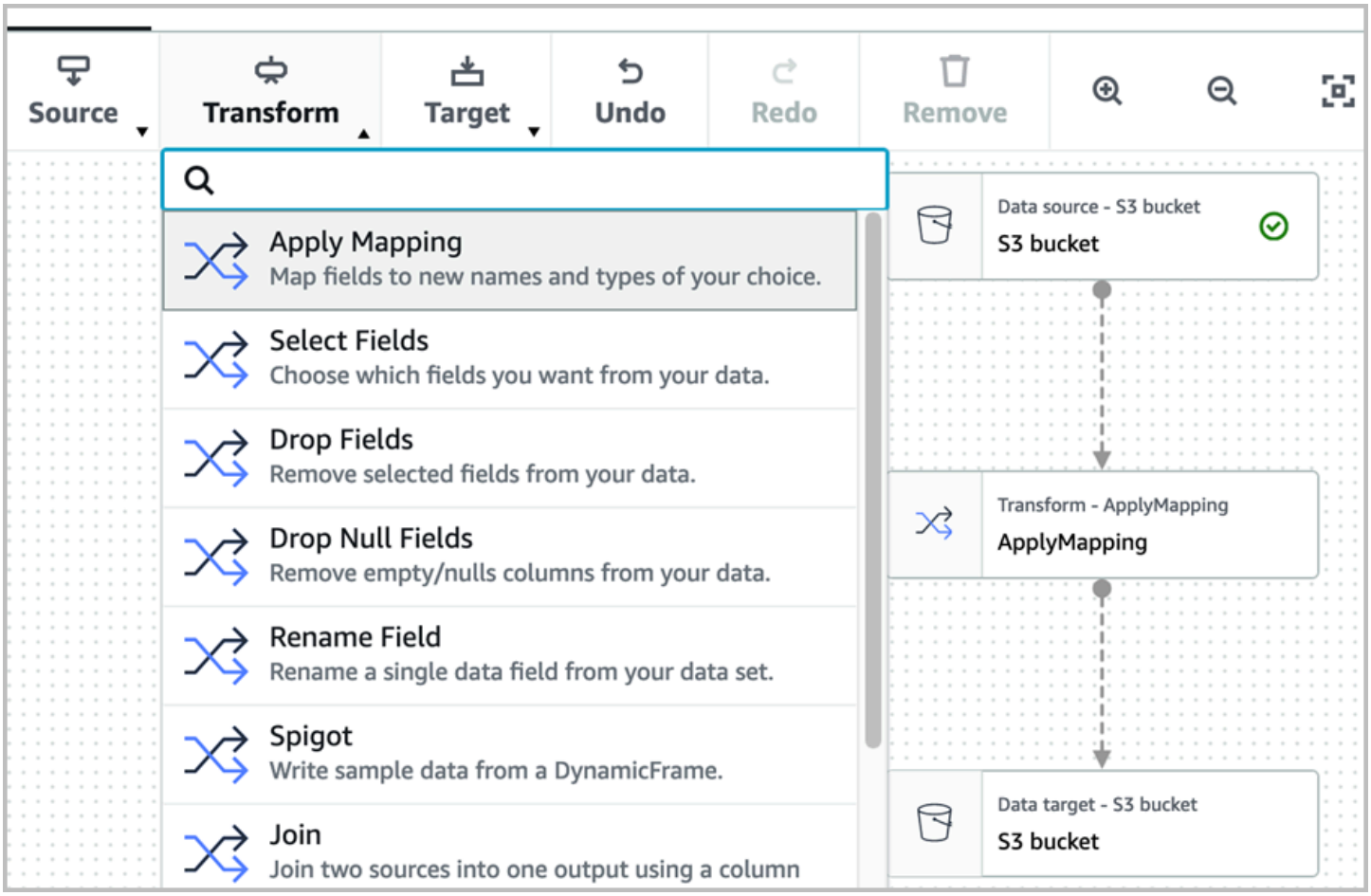


AWS SCT kann SSIS-Pakete in großen Mengen in AWS Glue Jobs konvertieren. Sie können das Skript bearbeiten, um bestehende Logik zu aktualisieren oder neue Logik hinzuzufügen, die auf Ihrem neuen Entwurf basiert. Wir empfehlen, dass Sie die Benennungskonventionen in den AWS SCT konvertierten Skripten befolgen, um die Skripten anzupassen.

Weitere Informationen finden Sie in der AWS SCT Dokumentation unter Konvertieren von SSIS AWS SCT in AWS Glue Using.

AWS Glue

AWS Glue Studio bietet eine grafische Oberfläche und ein Entwicklungserlebnis, das SSIS ähnelt, wie im folgenden Bildschirm dargestellt.



Wenn Sie keine grafische Oberfläche verwenden möchten, können Sie Ihre benutzerdefinierten Skripts auch mit den erforderlichen Python-Bibliotheken von der AWS Glue Konsole aus ausführen. Weitere Informationen finden Sie in der AWS Glue Dokumentation unter Bereitstellen eigener benutzerdefinierter Skripts.

AWS Glue stellt eine Reihe integrierter Transformationen für die Verarbeitung Ihrer Daten bereit. Diese ähneln SSIS-Datenflusstransformationen. Folgen Sie diesen bewährten Methoden, wenn Sie Ihre SSIS-ETL-Jobs migrieren, indem Sie Folgendes verwenden: AWS Glue

- Bereiten Sie eine Zuordnung von AWS Glue Transformationen zu den entsprechenden SSIS-Transformationen vor.
- Wenn Ihre Transformationen nicht AWS Glue Transformationen zugeordnet werden können, erstellen Sie sie mit einem benutzerdefinierten Python- oder Scala-Skript.
- Verwenden Sie für benutzerdefinierte Protokollierung (z. B. gelesene Zeilen, geschriebene Zeilen oder fehlerhafte Datensätze) zusätzlich zu Amazon benutzerdefinierte Skripts CloudWatch.

- Fügen Sie einen Entwicklungsendpunkt hinzu, um benutzerdefinierte Skripts lokal zu entwickeln und zu debuggen.

Amazon EMR

Sie können benutzerdefinierte Skripts (geschrieben in Python oder Scala) oder kompilierte Python-Bibliotheken in EMR-Clustern ausführen, wie bei. AWS Glue Folgen Sie diesen bewährten Methoden:

- Beginnen Sie mit speicheroptimierten Instance-Typen und erstellen Sie gleichzeitig EMR-Cluster mit dem Spark-Framework. (SSIS verwendet Speicherpuffer.)
- Erstellen Sie generische Python-Methoden, die jeder SSIS-Aufgabe oder -Transformation entsprechen. In der folgenden Abbildung erzeugt beispielsweise eine Methode, die zwei Datenrahmen als Eingabe verwendet, einen dritten Datenrahmen, der übereinstimmende Datensätze aus den beiden Datenrahmen als Ausgabe enthält. Dies funktioniert wie eine Transformation zum Zusammenführen von Verknüpfungen.

```
AWS: Add Debug Configuration | AWS: Edit Debug Configuration
def merge_join_spark(spark: SparkSession, df1: DataFrame, df2: DataFrame, join_type: str, join_column:str,
                     merge_fetch_columns: str):
    """Merge/Join directly
        spark: Current Spark session
        df1: First dataframe for merge join
        df2: Second dataframe for merge join
        join_type: Left/Inner/Outer join
        join_column : Column to be merge joined on
        merge_fetch_columns: Columns required in the output dataframe
    """
    try:

        spark.catalog.dropTempView("df1")
        spark.catalog.dropTempView("df2")
        df1.createOrReplaceTempView("df1")
        df2.createOrReplaceTempView("df2")
        merge_sql = f"select src.*, {merge_fetch_columns} from df1 {join_type} join df2 on df1.{join_column} = df2.{join_column}".format(
            merge_fetch_columns=merge_fetch_columns, join_type=join_type,join_column=join_column)
        logger.debug("Merge query is : " + merge_sql)
        output_df = spark.sql(merge_sql)
    except Exception as ex:
        logger.error("Merge Execution Failed for " + str(ex))
        raise Exception("Merge Execution Failed " + str(ex))
    finally:
        spark.catalog.dropTempView("df1")
        spark.catalog.dropTempView("df2")
    return output_df
```

Testen

Ein Testframework ist erforderlich, um die Vollständigkeit und Richtigkeit der Daten zu überprüfen. Dieses Framework sollte alle bestehenden Szenarien und alle Verbesserungen abdecken, die Sie bei der Migration Ihrer Jobs zu AWS vorgenommen haben.

- Überprüfung der Vollständigkeit:
  - Alle Jobs werden in ihren Zielstatus migriert.
  - Die gesamte Funktionalität wird in jedem Job migriert.
  - Alle Arten von Protokollen sind verfügbar, einschließlich Details zur Auftragsausführung, Fehlermeldungen, fehlerhaften Datensätzen und Zeilenanzahl.
- Überprüfung der Richtigkeit:
  - Die Qualität der Daten ist in den bestehenden und neuen Umgebungen konsistent.
  - Alle Spalten aller Tabellen stimmen überein, oder die Tabellen wurden verbessert AWS.
  - Alle Prüf- und Protokollierungsinformationen stimmen überein.

Sie sollten auch überprüfen, ob die Leistung Ihrer migrierten Jobs mit der Leistung Ihrer vorhandenen Jobs übereinstimmt.

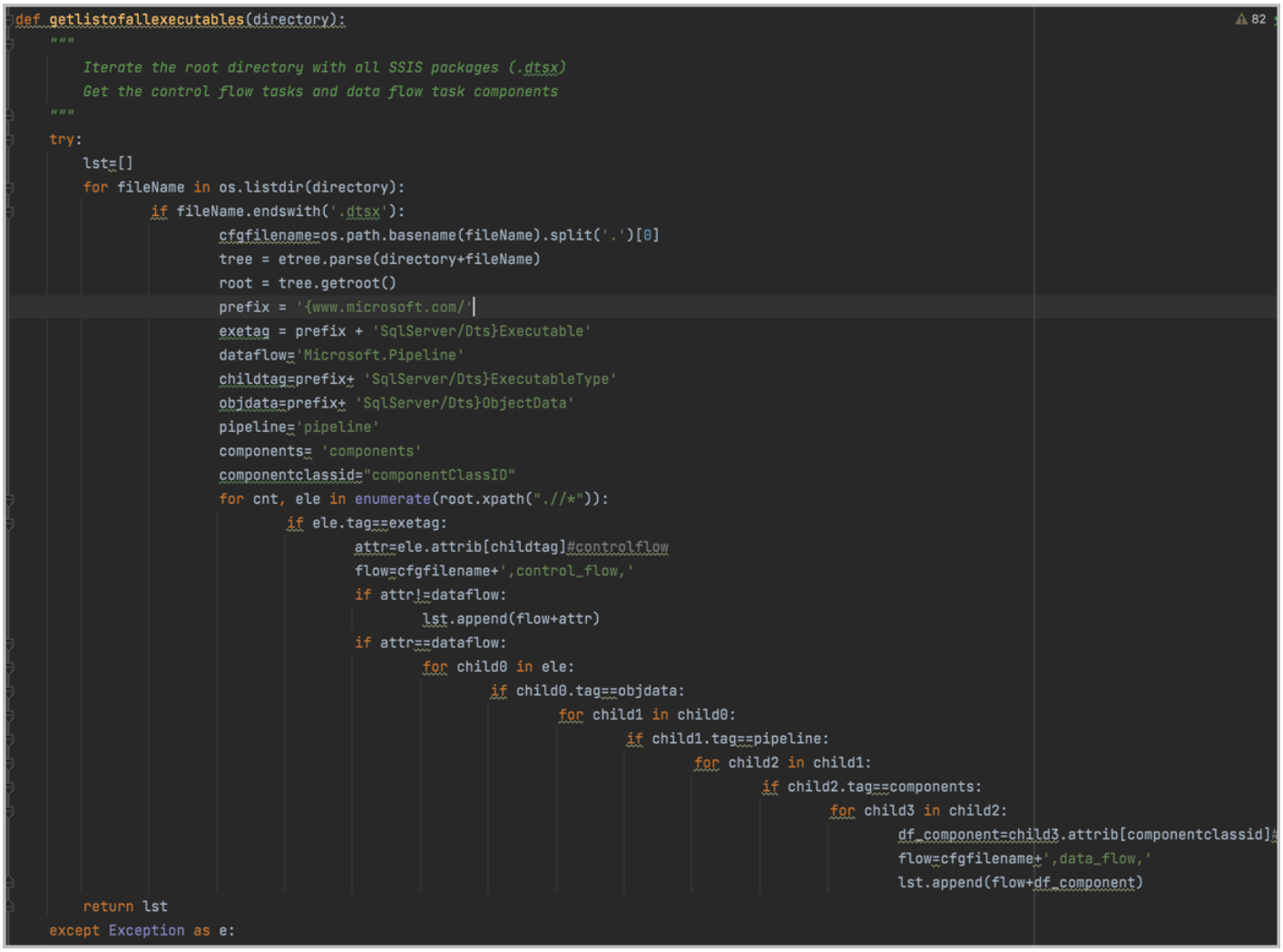
Bewährte Methoden

Halten Sie sich bei der Migration Ihrer SSIS-Jobs an die folgenden allgemeinen bewährten Methoden: AWS

- Wenn Sie Ihre Datenbanken zu einem AWS verwalteten Service wie Amazon Relational Database Service (Amazon RDS) migrieren, sollten Sie Wartungsaufgaben (wie Sicherung, Wiederherstellung, Aktualisierung von Statistiken oder Überprüfung der Datenbankintegrität), die nicht relevant sind oder von nicht verwaltet werden, abgrenzen. AWS
- Erstellen Sie wiederverwendbare Skripts und Methoden, um die Entwicklung zu standardisieren und den Aufwand zu reduzieren.
- Testen Sie migrierte Jobs immer auf Infrastruktur und Datenvolumen, die den Produktionsressourcen entsprechen.
- SSIS-Pakete sind im XML-Format. Erstellen Sie XML-Parser-Dienstprogramme, um Migrationsaktivitäten nach Möglichkeit zu automatisieren. Das folgende Beispiel zeigt ein SSIS-Paket im XML-Format.

```
    DTS:LastModifiedProductVersion="11.0.2100.60"
    DTS:LocaleID="1033"
    DTS:ObjectName="Package"
    DTS:PackageType="5"
    DTS:VersionBuild="1"
    DTS:VersionGUID="{EC16490E-6783-4BE6-92CA-FDD5BC644F88}">
    <DTS:Property
      DTS:Name="PackageFormatVersion">6</DTS:Property>
    <DTS:ConnectionManagers>
      <DTS:ConnectionManager
```

Die folgende Abbildung zeigt ein Beispiel für einen XML-Parser.



```
def getlistofallexecutables(directory):
    """
        Iterate the root directory with all SSIS packages (.dtsx)
        Get the control flow tasks and data flow task components
    """
    try:
        lst=[]
        for fileName in os.listdir(directory):
                if fileName.endswith('.dtsx'):
                        cfgfilename=os.path.basename(fileName).split('.')[0]
                        tree = etree.parse(directory+fileName)
                        root = tree.getroot()
                        prefix = '{www.microsoft.com/'
                        exetag = prefix + 'SqlServer/Dts}Executable'
                        dataflow='Microsoft.Pipeline'
                        childtag=prefix+ 'SqlServer/Dts}ExecutableType'
                        objdata=prefix+ 'SqlServer/Dts}ObjectData'
                        pipeline='pipeline'
                        components= 'components'
                        componentclassid="componentClassID"
                        for cnt, ele in enumerate(root.xpath(".//*")):
                                if ele.tag==exetag:
                                        attr=ele.attrib[childtag]#controlflow
                                        flow=cfgfilename+',control_flow,'
                                        if attr!=dataflow:
                                                lst.append(flow+attr)
                                        if attr==dataflow:
                                                for child0 in ele:
                                                        if child0.tag==objdata:
                                                                for child1 in child0:
                                                                        if child1.tag==pipeline:
                                                                                for child2 in child1:
                                                                                        if child2.tag==components:
                                                                                                for child3 in child2:
                                                                                                        df_component=child3.attrib[componentclassid]
                                                                                                        flow=cfgfilename+',data_flow,'
                                                                                                        lst.append(flow+df_component)
        return lst
    except Exception as e:
```

- Verwenden Sie Prüfsummen und Hashwerte, um Richtigkeit und Vollständigkeit zu testen.
- Wenn Sie benutzerdefinierte Bibliotheken erstellen, implementieren Sie Threading, um Aufgaben parallel auszuführen. Dies verbessert die Arbeitsleistung.
- Nehmen Sie die bestehende Umgebung erst dann außer Betrieb, wenn alle Arbeitsplätze AWS für einen bestimmten Zeitraum stabil funktionieren.
- Verwenden Sie Amazon CloudWatch für die Protokollierung, um den Aufwand für die Anpassung der Protokollierung zu reduzieren.
- Verwenden Sie Amazon Simple Notification Service (Amazon SNS), um benutzerdefinierte Aufgaben für das Senden von Benachrichtigungen zu ersetzen. Verwenden Sie Amazon Simple Email Service (Amazon SES), um benutzerdefinierte E-Mail-Aufgaben zu ersetzen.

Häufig gestellte Fragen

Dieser Abschnitt enthält Antworten auf häufig gestellte Fragen zur Migration Ihrer SSIS-ETL-Jobs zu. AWS

Sollte ich die SSIS-Jobs während der Migration verbessern?

Ja. Implementieren Sie alle wichtigen Verbesserungen oder Fehlerkorrekturen in bestehenden und neuen Jobs gleichzeitig. Änderungen mit niedriger Priorität können nach der Migration implementiert werden.

Wie kann ich Jobs überwachen? AWS

Sie können Amazon verwenden CloudWatch, um Jobs zu überwachen und regelbasierte Benachrichtigungen zu senden.

Wann kann ich meine lokalen SSIS-Jobs außer Betrieb nehmen?

Es wird empfohlen, alte und neue Jobs für einen bestimmten Zeitraum parallel zu verwalten, bis die Leistung, Genauigkeit und Vollständigkeit der Daten in beiden Umgebungen identisch sind. Sie können auch Berichtssysteme replizieren, um Berichte aus beiden Umgebungen miteinander zu verbinden und zu vergleichen. Sie können Ihre SSIS-Jobs außer Betrieb nehmen, wenn die Berichte identisch sind.

Nächste Schritte

Als nächsten Schritt empfehlen wir Ihnen, einen Machbarkeitsnachweis zu erstellen, um Ihren Bereitstellungsansatz für die Zielarchitektur festzulegen. Für Ihren Machbarkeitsnachweis:

- Erstellen Sie AWS Glue Jobs mithilfe der grafischen Benutzeroberfläche von AWS Glue Studio.
- Verwenden Sie AWS Schema Conversion Tool (AWS SCT), um AWS Glue Skripts aus SSIS-Paketen zu generieren.
- Erstellen Sie mithilfe von Spark-Bibliotheken ein benutzerdefiniertes Migrationsframework.
- Erstellen Sie ein Test-Framework.
- Identifizieren Sie Tools und Prozesse für die kontinuierliche Integration und kontinuierliche Bereitstellung (CI/CD).

Zugehörige Ressourcen

- Umwandlung von SSIS in die Verwendung von AWS GlueAWS SCT
- Amazon RDS-Backup und -Wiederherstellung
- AWS Glue Dokumentation
- AWS Glue FAQs
- Kontinuierliche Protokollierung von AWS Glue Jobs
- Amazon EMR-Dokumentation
- Amazon EMR FAQs
- Überwachen Sie Metriken mit CloudWatch
- Beschleunigen Sie die Validierung umfangreicher Datenmigrationen mit PyDeequ
- AWS Kompetenzpartner für Daten und Analysen

Dokumentverlauf

In der folgenden Tabelle werden wichtige Änderungen in diesem Leitfaden beschrieben. Um Benachrichtigungen über zukünftige Aktualisierungen zu erhalten, können Sie einen RSS-Feed abonnieren.

Änderung	Beschreibung	Datum
Erste Veröffentlichung	—	6. Dezember 2021

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.