aws

Guide du développeur de base de données

Amazon Redshift



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

Amazon Redshift: Guide du développeur de base de données

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

Les marques et la présentation commerciale d’Amazon ne peuvent être utilisées en relation avec un produit ou un service qui n’est pas d’Amazon, d’une manière susceptible de créer une confusion parmi les clients, ou d’une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d’Amazon appartiennent à leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

Introduction 1
Conditions préalables à l'utilisation d'Amazon Redshift 1
Architecture d'Amazon Redshift 2
Architecture d'entrepôt de données 2
Performances 6
Stockage en colonnes 9
Gestion de la charge de travail 12
Utilisation d’Amazon Redshift avec d’autres services 13
Exemple de base de données 14
Bonnes pratiques 22
Procéder à une preuve de concept 23
Étape 1 : Déterminez le périmètre de votre POC 23
Étape 2 : Lancez Amazon Redshift 25
Étape 3 : Chargez vos données 26
Étape 4 : Analysez vos données 27
Étape 5 : Optimisation 30
Bonnes pratiques pour la conception de tables 31
Choix de la meilleure clé de tri 31
Choisir le meilleur style de distribution 32
Utiliser la compression automatique 33
Définir les contraintes 34
Utiliser la plus petite taille de colonne possible 35
Utiliser les types de données date/time pour les colonnes de date 35
Bonnes pratiques pour le chargement des données 35
Découvrez comment charger des données 36
Utiliser une commande COPY pour charger les données 36
Utiliser une seule commande COPY 36
Chargement de fichiers de données 37
Compression de vos fichiers de données 38
Vérifier les fichiers de données avant et après un chargement 38
Utiliser une insertion multiligne 38
Utiliser une insertion en bloc 39
Charger les données dans l'ordre de la clé de tri 39
Charger les données en blocs séquentiels 40

Utiliser les tables chronologiques 40
Planifier la maintenance Windows 41
Bonnes pratiques pour la conception des requêtes 41
Suivez les recommandations des conseillers 43
Régions Amazon Redshift dans lesquelles Advisor est pris en charge 44
Consulter les recommandations des conseillers 45
Recommandations Advisor 46
Didacticiels 63
Optimisation automatique des tables 65
Activation, désactivation et surveillance de l'optimisation automatique des tables 66
Activation de l’optimisation automatique des tables 66
Désactivation de l’optimisation automatique d’une table 66
Surveillance de l'optimisation automatique des tables 67
Compression de colonnes 68
encodages de compression 69
Test des encodages de compression 80
Distribution de données 87
Concepts de distribution de données 88
Styles de distribution 90
Affichage des styles de distribution 91
Évaluation des modèles de requêtes 93
Détermination des styles de distribution 94
Évaluation du plan de requête 95
Exemple de plan de requête 97
Exemples de distribution 102
Clés de tri 104
Tri multidimensionnel de la disposition des données (version préliminaire) 106
Clé de tri composée 107
Clé de tri entrelacée 108
Contraintes liées aux tables 109
Chargement des données 111
Chargement de tables avec COPY 115
Informations d’identification et autorisations d’accès 117
Préparation de vos données d’entrée 119
Chargement des données à partir d’Amazon S3 120
Chargement de données à partir d’Amazon EMR 134

Chargement des données à partir des hôtes distants 140
Chargement à partir d'Amazon DynamoDB 150
Vérification que les données ont été chargées correctement 153
Validation des données d'entrée 154
Chargement des tables avec compression automatique 154
Optimisation des tables étroites 157
Chargement des valeurs par défaut des colonnes 158
Résolution des problèmes de chargement de données 159
Créez une intégration d'événements S3 pour copier automatiquement des fichiers 167
Prérequis 168
Création d'une intégration d'événements S3 172
Création et suivi d'un COPY JOB 175
Considérations 177
Chargement de tables avec DML 177
Mise à jour et insertion 178
Exécution d'une copie complète 186
Analyse des tables 191
Analyse automatique 191
Analyse des données des nouvelles tables 191
Historique de la commande ANALYZE 197
Exécution de l'opération VACUUM sur les tables 199
Tri automatique des tables 199
Suppression de vide automatique 200
Fréquence de VACUUM 201
Phase de tri et phase de fusion 201
Seuil de VACUUM 202
Types d'opération VACUUM 202
Minimiser les temps d'aspiration 202
Gestion des opérations d'écriture simultanées 211
Isolement sérialisable 212
Écriture et read/write opérations 218
Exemples d'écritures simultanées 220
Didacticiel : chargement des données à partir d'Amazon S3 223
Prérequis 223
Présentation 224
Étape 1 : créer un cluster 224

Étape 2 : Télécharger les fichiers de données ... 225
Étape 3 : Charger les fichiers dans un compartiment Amazon S3 ... 226
Étape 4 : Créer des exemples de tables ... 228
Étape 5 : Exécuter les commandes COPY ... 231
Étape 6 : Vider et analyser la base de données ... 249
Étape 7 : Nettoyer vos ressources ... 250
Récapitulatif ... 250
Déchargement des données ... 252
Déchargement de données vers Amazon S3 ... 252
Déchargement de fichiers de données chiffrés ... 256
Déchargement de données au format délimité ou au format à largeur fixe ... 258
Rechargement de données déchargées ... 259
Fonctions définies par l'utilisateur ... 261
Sécurité et autorisations UDF ... 262
Prévention des conflits de dénomination des fonctions UDF ... 262
Surcharge des noms de fonctions ... 263
Éviter les conflits avec les fonctions intégrées d'Amazon Redshift ... 263
SQL scalaire UDFs ... 264
exemple ... 264
Python scalaire UDFs ... 265
exemple ... 266
Types de données de fonctions Python définies par l'utilisateur ... 267
Prise en charge du langage Python ... 268
Constraints ... 273
Erreurs et avertissements de journalisation ... 274
Lambda scalaire UDFs ... 275
Gestion de la sécurité et des autorisations Lambda UDF ... 277
Configuration du paramètre d'autorisation pour Lambda UDFs ... 278
Utilisation de l'interface JSON entre Amazon Redshift et Lambda ... 280
Exemples de cas d'utilisation pour UDFs ... 283
Création de procédures stockées ... 284
Présentation des procédures stockées ... 285
Dénomination des procédures stockées ... 288
Sécurité et privilèges ... 289
Retour d’un ensemble de résultats ... 290
Gestion des transactions ... 293

Interception des erreurs 306
Enregistrement des procédures stockées 315
Limites 315
Guide de référence du langage PL/pgSQL 316
Conventions de référence du langage PL/pgSQL 317
Structure de PL/pgSQL 318
Instructions PL/pgSQL prises en charge 324
Vues matérialisées 341
Requêtes de vue matérialisées 344
Réécriture automatique des requêtes pour utiliser des vues matérialisées 345
Notes d'utilisation 346
Comment fonctionne la réécriture automatique des requêtes des vues matérialisées 347
Limites 347
Vues matérialisées sur des tables de lacs de données externes 348
Limites 349
Actualisation d'une vue matérialisée 350
Actualisation automatique d'une vue matérialisée 353
Vues matérialisées automatisées 354
Portée et considérations SQL pour les vues matérialisées automatisées 356
Vues matérialisées automatisées 357
Facturation pour les vues matérialisées automatisées 358
Ressources supplémentaires 358
Utilisation d'une fonction définie par l'utilisateur (UDF) dans une vue matérialisée 358
Référencer un UDF dans une vue matérialisée 358
Diffusion de l'ingestion vers une vue matérialisée 360
Comment les données circulent d'un service de streaming vers Redshift 361
Meilleures pratiques d'analyse des données pour améliorer les performances 361
Comportement d'ingestion du streaming et types de données 362
Mise en route de l'ingestion en streaming à partir d'Amazon Kinesis Data Streams 368
Débuter avec l'ingestion du streaming depuis Apache Kafka 373
Didacticiel sur l'ingestion en streaming de données de bornes de véhicules électriques avec Kinesis 389
Vues du catalogue de données 393
Prérequis 394
End-to-end exemple 395
Journalisation sécurisée 396

Considérations ... 402
Interrogation des données spatiales ... 404
Didacticiel : Utiliser les fonctions SQL spatiales ... 408
Prérequis ... 408
Étape 1 : Créer des tables et charger des données de test ... 409
Étape 2 : Interroger les données spatiales ... 412
Étape 3 : Nettoyer vos ressources ... 416
Chargement d'un shapefile ... 416
Terminologie ... 418
Cadre de délimitation ... 418
Validité géométrique ... 418
Simplicité géométrique ... 421
H3 ... 423
Considérations ... 424
Interrogation de données avec requête fédérée ... 426
Commencer à utiliser les requêtes fédérées vers PostgreSQL ... 427
Commencer à utiliser des requêtes fédérées dans PostgreSQL avec CloudFormation ... 428
Lancement d'une CloudFormation pile pour les requêtes fédérées Redshift ... 429
Interrogation de données à partir du schéma externe ... 431
Commencer à utiliser des requêtes fédérées vers MySQL ... 432
Création d'un secret et d'un rôle IAM ... 433
Prérequis ... 433
Exemples d'utilisation d'une requête fédérée ... 436
Exemple d'utilisation d'une requête fédérée avec PostgreSQL ... 436
Exemple d'utilisation d'un nom en casse mixte ... 438
Exemple d'utilisation d'une requête fédérée avec MySQL ... 440
Différences de type de données ... 441
Considérations ... 445
Versions prises en charge des bases de données fédérées ... 447
Amazon Redshift Spectrum ... 449
Présentation d'Amazon Redshift Spectrum ... 449
Régions Amazon Redshift Spectrum ... 451
Limites d'Amazon Redshift Spectrum ... 451
Mise en route avec Amazon Redshift Spectrum ... 452
Prérequis ... 453
CloudFormation ... 454

Mise en route avec Amazon Redshift Spectrum étape par étape 454
Étape 1. Créer un rôle IAM 454
Étape 2 : Association du rôle IAM au cluster 458
Étape 3 : Création d'un schéma externe et d'une table externe 459
Étape 4 : Interrogation de vos données dans Amazon S3 461
Lancez votre CloudFormation stack puis interrogez vos données 464
Politiques IAM pour Amazon Redshift Spectrum 468
Autorisations Amazon S3 468
Autorisations Amazon S3 entre comptes 469
Accorder ou restreindre l'accès l'aide de Redshift Spectrum 470
Autorisations minimales 471
Création de chaînes de rôles IAM 472
Accès aux AWS Glue données 473
Spectre de Redshift et Lake Formation 482
Utilisation de filtres de données pour la sécurité au niveau de la ligne et au niveau de la cellule 484
Fichiers de données pour les requêtes dans Amazon Redshift Spectrum 484
Formats de données pour Redshift Spectrum 485
Types de compression pour Redshift Spectrum 486
Chiffrement pour Redshift Spectrum 487
Schémas externes 488
Utilisation des catalogues externes 490
Tables externes 495
Pseudocolonnes 497
Partitionnement des tables externes Redshift Spectrum 498
Mappage des colonnes ORC 504
Créer des tables externes pour les données gérées par Hudi 507
Créer des tables externes pour les données Delta Lake 509
Utilisation des tables Apache Iceberg 511
Éléments à prendre en considération lors de l'utilisation de tables Apache Iceberg 512
Types de données pris en charge 514
Performances des requêtes Amazon Redshift Spectrum 516
Options de traitement des données 520
Exécution de sous-requêtes corrélées 520
Métriques 521
Résolution des problèmes de requêtes 522

Dépassement des nouvelles tentatives 523
Accès limité 523
Limite de ressource dépassée 525
Aucune ligne retournée pour une table partitionnée 525
Erreur Non autorisé 525
Formats de données incompatibles 526
Erreur de syntaxe lors de l'utilisation de Hive DDL dans Amazon Redshift 526
Autorisation de créer des tables temporaires 527
Plage non valide 527
Numéro de version Parquet non valide 527
Didacticiel : Faire une requête de données imbriquées avec Amazon Redshift Spectrum 528
Présentation 528
Étape 1 : Création d'un tableau externe contenant des données imbriquées 529
Étape 2 : Faire une requête de vos données imbriquées dans Amazon S3 avec des extensions SQL 530
Cas d'utilisation de données imbriquées 535
Limitations des données imbriquées (version préliminaire) 537
Sérialisation de JSON imbriqué complexe 540
HyperLogLog croquis 543
Considérations 544
Limites 545
Exemples 546
Exemple : renvoyer la cardinalité dans une sous-requête 546
Exemple : renvoyer un type HLLSKETCH à partir d'esquisses combinées dans une sous-requête 547
Exemple : renvoyer une HyperLogLog esquisse en combinant plusieurs esquisses 547
Exemple : générer des HyperLogLog esquisses sur des données S3 à l'aide de tables externes 548
Requêtes entre bases de données 552
Considérations 555
Limites 555
Exemples 556
Utilisation de requêtes entre bases de données avec l'éditeur de requêtes 562
Compatibilité avec Apache Iceberg 564
Régions dans lesquelles la compatibilité avec Apache Iceberg est disponible 564

Considérations et limites relatives à l'utilisation des catalogues Amazon Redshift dans le AWS Glue Data Catalog ... 565
Exigences de la politique IAM ... 566
Enregistrement de clusters et d'espaces de noms ... 567
Annulation de l'enregistrement des clusters et des espaces de noms ... 568
Groupes de travail gérés ... 570
Consultation des catalogues enregistrés dans le AWS Glue Data Catalog ... 570
Partage des données ... 573
Cas d’utilisation du partage de données ... 573
Plusieurs entrepôts écrivent des cas d'utilisation ... 574
Partage de données à différents niveaux ... 575
Gestion de la cohérence du partage des données ... 575
Considérations ... 575
... 575
Considérations générales relatives au partage de données ... 576
Considérations relatives au partage de données en lecture et en écriture ... 578
Considérations relatives au partage de données à l'aide de tables de lacs de données ... 579
Considérations relatives au partage de données avec AWS Lake Formation ... 580
Considérations relatives au partage de données avec AWS Data Exchange ... 581
Autorisations que vous pouvez accorder aux unités de partages des données ... 582
Instructions SQL prises en charge pour le partage de données et écritures destinées aux consommateurs ... 583
Des instructions SQL non prises en charge pour le partage de données s'adressent aux consommateurs ... 584
Disponible Régions AWS ... 585
Premiers pas ... 589
Commencer à partager des données en lecture seule dans la console ... 589
Commencer à partager des données en lecture seule avec l'interface SQL ... 610
Débuter avec les écritures dans plusieurs entrepôts ... 659
Commencer avec CloudFormation ... 696
Types de partages de données ... 702
Unités de partage des données standard ... 703
AWS Data Exchange partages de données ... 704
Unités de partage des données gérées par AWS Lake Formation ... 708
État du partage de données ... 709
Gérer l’accès aux opérations d’API d’unités de partage de données avec les politiques IAM ... 711

Connexion aux bases de données des consommateurs 713
Accéder aux données partagées 713
Accès aux métadonnées pour les objets partagés 714
Intégrer le partage de données Amazon Redshift aux outils de business intelligence 715
Surveillance et audit du partage des données 715
Intégrer le partage de données Amazon Redshift avec AWS CloudTrail 717
Données semi-structurées dans Amazon Redshift 718
Cas d'utilisation pour le type de données SUPER 718
Concepts pour l'utilisation des types de données SUPER 720
Considérations relatives aux données Super 722
Jeu de données échantillon SUPER 722
Chargement de données semi-structurées dans Amazon Redshift 725
Analyse de documents JSON en colonnes SUPER 725
Utilisation de COPY pour charger des données JSON dans Amazon Redshift 726
Déchargement des données semi-structurées 731
Interrogation de données semi-structurées 732
Navigation 733
Désimbriquer des requêtes 734
Dépivotement d'objet 736
Typage dynamique 737
Sémantique laxiste 741
Types d'introspection 741
Classer par 743
Opérateurs et fonctions 744
Opérateurs arithmétiques 744
Fonctions arithmétiques 745
Fonctions de tableau 745
Configurations SUPER 747
Modes laxistes et stricts pour SUPER 747
Accès aux champs JSON avec des lettres majuscules et à casse mixte 748
Options d'analyse 750
Limites 750
Type de données SUPER et vues matérialisées 754
Accélération des requêtes PartiQL 754
Limites de l'utilisation du type de données SUPER avec les vues matérialisées 758
Machine learning 760

Présentation du machine learning 762
Comment le machine learning peut résoudre un problème 763
Termes et concepts pour Amazon Redshift ML 765
Machine learning pour les novices et les experts 766
Coûts d'utilisation d'Amazon Redshift ML 769
Coûts liés à l'utilisation d'Amazon Redshift ML avec l'IA SageMaker 769
Coûts d'utilisation d'Amazon Redshift ML avec Amazon Bedrock 769
Mise en route : Amazon Redshift ML 771
Configuration administrative 772
Utilisation de l'explicabilité du modèle avec Amazon Redshift ML 778
Métriques de probabilité Amazon Redshift ML 779
Tutoriels pour Amazon Redshift ML 782
Tutoriel : Créer des modèles de désabonnement des clients 783
Tutoriel : Création de modèles de clustering en k-moyennes 792
Tutoriel : Création de modèles de classification multiclasse 802
Tutoriel : Construire XGBoost des modèles 812
Tutoriel : Création de modèles de régression 818
Tutoriel : Création de modèles de régression avec apprentissage linéaire 831
Tutoriel : Création de modèles de classification multiclasse avec apprentissage linéaire 839
Intégration d'Amazon Redshift ML à Amazon Bedrock 862
Création d’un rôle IAM 862
Création d'un modèle externe 863
Utilisation d'un modèle externe 864
Ingénierie rapide 866
Réglage des performances des requêtes 868
Traitement des requêtes 868
Workflow d’exécution et de planification de requête 869
Création et interprétation d'un plan de requêtes 871
Révision des étapes du plan de requête 880
Facteurs affectant la performance des requêtes 882
Analyse et amélioration des requêtes 884
Flux de travail d’analyse des requêtes 885
Révision des alertes de requêtes 886
Analyse du plan de requête 888
Analyse du résumé de la requête 889
Amélioration des requêtes 897

Requêtes de diagnostics pour l'ajustement des requêtes 902
Résolution des problèmes de requêtes 906
Échecs des connexions 907
La requête se bloque 908
La requête est trop longue 909
La charge échoue 911
La charge est trop longue 911
Le chargement des données est incorrect 912
Définition du paramètre de taille d'extraction JDBC 912
Gestion de la charge de travail 914
Commutation du mode WLM 916
Modifier la configuration WLM 916
Migration de la gestion manuelle de la charge de travail à la gestion automatique de la charge de travail 917
Gestion automatique de la charge de travail 919
Priorité 920
Mode de mise à l'échelle de la simultanéité 920
Groupes d'utilisateurs 921
Rôles utilisateurs 921
Groupes de requêtes 921
Caractères génériques 922
Règles de surveillance de requête 922
Vérification de la gestion automatique de la charge de travail 922
Priorité de requête 923
Gestion manuelle de la charge de travail 928
Mode de mise à l'échelle de la simultanéité 930
Niveau de simultanéité 930
Groupes d'utilisateurs 933
Rôles utilisateurs 933
Groupes de requêtes 933
Caractères génériques 934
Pourcentage de mémoire WLM à utiliser 934
Délai WLM 935
Règles de surveillance de requête 935
Saut de file d'attente des requêtes WLM 936
Didacticiel : Configuration des files d'attente de gestion manuelle de la charge de travail 940

Mise à l'échelle de la simultanéité 959
Capacités de mise à l'échelle de la simultanéité 960
Limitations de la mise à l'échelle de la simultanéité 961
Régions pour la mise à l'échelle de la simultanéité 962
Candidats à la mise à l'échelle de la simultanéité 963
Configuration de files d'attente de mise à l'échelle de la simultanéité 925
Surveillance de la mise à l'échelle de la simultanéité 964
Vues système de la mise à l'échelle de la simultanéité 965
Accélération des requêtes courtes 966
Durée d'exécution de SQA maximale 967
Surveillance de la SQA 968
Règles d'affectation de file d'attente de WLM 969
Exemple d'affectations de files d'attente 971
Affectation des requêtes à des files d'attente 973
Affectation des requêtes aux files d'attente en fonction des rôles utilisateurs 973
Affectation des requêtes aux files d'attente en fonction des groupes d'utilisateurs 974
Affectation d'une requête à un groupe de requêtes 974
Affectation des requêtes à la file d'attente Super-utilisateur 975
Propriétés dynamiques et statiques 976
Allocation de mémoire dynamique WLM 978
Exemple WLM dynamique 979
Règles de surveillance de requête 981
Définition d'une règle de surveillance de requête 982
Métriques de surveillance des requêtes pour cluster Amazon Redshift provisionné 984
Métriques de surveillance des requêtes pour Amazon Redshift sans serveur 989
Modèles de règles de surveillance de requête 990
Tables et vues système pour les règles de surveillance de requête 992
Tables et vues système WLM 992
Classe de service WLM IDs 994
Sécurité des bases de données 996
Présentation de la sécurité d'Amazon Redshift 997
Autorisations par défaut des utilisateurs de base de données 998
Super-utilisateurs 999
Users 1000
Création, modification et suppression d'utilisateurs 1001
Groups 1001

Création, modification et suppression de groupes ... 1002
Exemple de contrôle d'accès utilisateur et de groupe ... 1002
Schémas ... 1004
Chemin de recherche ... 1005
Création, modification et suppression de schémas ... 1005
Autorisations ... 1006
Contrôle d'accès basé sur les rôles ... 1006
Hiérarchie des rôles ... 1007
Attribution des rôles ... 1008
Rôles définis par le système Amazon Redshift ... 1008
Autorisations système ... 1013
Autorisations d'objet de base de données ... 1030
Instruction ALTER DEFAULT PRIVILEGES pour le RBAC ... 1030
Considérations concernant l'utilisation des rôles ... 1030
Gestion des rôles ... 1031
Tutoriel : Création de rôles et interrogation avec RBAC ... 1032
Sécurité au niveau des lignes ... 1052
Utilisation des politiques RLS dans les instructions SQL ... 1053
Association de plusieurs politiques par utilisateur ... 1053
Propriété et gestion des politiques RLS ... 1055
Objets et principes dépendants des politiques ... 1057
Considérations et restrictions ... 1059
Bonnes pratiques ... 1063
End-to-end exemple ... 1065
Sécurité des métadonnées ... 1070
Masquage dynamique des données ... 1071
Commandes SQL pour les politiques DDM ... 1072
Hiérarchie des politiques DDM ... 1073
Utilisation du DDM avec des chemins de type SUPER ... 1075
Masquage conditionnel des données dynamiques ... 1080
Vues du système DDM ... 1082
Considérations ... 1084
End-to-end exemple ... 1089
Autorisations étendues ... 1092
Considérations ... 1092
Référence SQL ... 1094

Amazon Redshift SQL 1094
Fonctions SQL prises en charge sur le nœud principal 1095
Amazon Redshift et PostgreSQL 1097
Utilisation de SQL 1106
Conventions du guide de référence SQL 1106
Éléments de base 1107
Expressions 1165
Conditions 1170
Commandes SQL 1199
ABORT 1204
ALTER DATABASE 1205
ALTER DATASHARE 1211
ALTER DEFAULT PRIVILEGES 1215
MODIFIER LE SCHÉMA EXTERNE 1219
MODIFIER LA VUE EXTÉRIEURE 1222
ALTER FUNCTION 1223
ALTER GROUP 1224
ALTER IDENTITY PROVIDER 1226
MODIFIER LA POLITIQUE DE MASQUAGE 1229
ALTER MATERIALIZED VIEW 1230
ALTER RLS POLICY 1237
ALTER ROLE 1238
ALTER PROCEDURE 1240
ALTER SCHEMA 1241
ALTER SYSTEM 1243
ALTER TABLE 1245
ALTER TABLE APPEND 1272
ALTER USER 1278
ANALYSE 1285
ANALYZE COMPRESSION 1288
ATTACH MASKING POLICY 1291
ATTACH RLS POLICY 1293
BEGIN 1294
CALL 1297
ANNULER 1300
CLOSE 1303

COMMENT 1303
COMMIT 1306
COPY 1307
CREATE DATABASE 1417
CREATE DATASHARE 1436
CREATE EXTERNAL FUNCTION 1438
CRÉER UN MODÈLE EXTERNE 1449
CREATE EXTERNAL SCHEMA 1454
CREATE EXTERNAL TABLE 1466
CRÉER UNE VUE EXTERNE 1496
CREATE FUNCTION 1497
CREATE GROUP 1504
CREATE IDENTITY PROVIDER 1505
CREATE LIBRARY 1508
CREATE MASKING POLICY 1512
CREATE MATERIALIZED VIEW 1513
CREATE MODEL 1519
CREATE PROCEDURE 1550
CREATE RLS POLICY 1557
CREATE ROLE 1559
CREATE SCHEMA 1560
CREATE TABLE 1564
CREATE TABLE AS 1589
CREATE USER 1602
CREATE VIEW 1610
DEALLOCATE 1615
DECLARE 1616
DELETE 1621
DESC DATASHARE 1624
DESC IDENTITY PROVIDER 1626
DETACH MASKING POLICY 1627
DETACH RLS POLICY 1628
DROP DATABASE 1629
DROP DATASHARE 1630
SUPPRIMER LA VUE EXTÉRIEURE 1632
DROP FUNCTION 1633

DROP GROUP 1635
DROP IDENTITY PROVIDER 1636
DROP LIBRARY 1637
DROP MASKING POLICY 1637
DROP MODEL 1638
DROP MATERIALIZED VIEW 1639
DROP PROCEDURE 1640
DROP RLS POLICY 1641
DROP ROLE 1642
DROP SCHEMA 1644
DROP TABLE 1646
DROP USER 1650
DROP VIEW 1652
FIN 1655
EXECUTE 1656
EXPLAIN 1657
FETCH 1666
GRANT 1668
INSERT 1696
INSERT (table externe) 1704
LOCK 1706
MERGE 1707
PREPARE 1715
REFRESH MATERIALIZED VIEW 1717
RESET 1721
REVOKE 1722
ROLLBACK 1742
SELECT 1743
SELECT INTO 1822
SET 1823
SET SESSION AUTHORIZATION 1829
SET SESSION CHARACTERISTICS 1830
MONTRER 1830
SHOW COLUMNS 1831
SHOW EXTERNAL TABLE 1834
SHOW DATABASES 1837

AFFICHER LES SUBVENTIONS 1841
SHOW MODEL 1847
SHOW DATASHARES 1850
SHOW PROCEDURE 1852
SHOW SCHEMAS 1853
SHOW TABLE 1855
SHOW TABLES 1857
SHOW VIEW 1859
START TRANSACTION 1860
TRUNCATE 1860
UNLOAD 1862
UPDATE 1898
USE 1907
VACUUM 1917
Référence sur les fonctions SQL 1925
Fonctions exécutées uniquement sur le nœud principal 1926
Fonctions d'agrégation 1928
Fonctions de tableau 1958
Fonctions d'agrégation bit par bit 1963
Expressions conditionnelles 1971
Fonctions de formatage des types de données 1987
Fonctions de date et d'heure 2026
Fonctions de hachage 2100
HyperLogLog fonctions 2110
Fonctions JSON 2116
Solutions de machine learning 2138
Fonctions mathématiques 2141
Fonctions d'objet 2181
Fonctions spatiales 2191
Fonctions de chaîne 2338
Fonctions d'informations sur le type SUPER 2419
Fonctions VARBYTE 2435
Fonctions de fenêtrage 2444
Fonctions d'administration système 2513
Fonctions d'informations système 2525
Mots réservés 2558

Informations de référence sur les tables et les vues système 2563
Types de tables et de vues système 2564
Visibilité des données dans les tables et vues système 2565
Filtrage des requêtes générées par le système 2566
Migration de requêtes provisionnées uniquement vers des requêtes SYS Monitoring View 2566
Migration à partir de clusters provisionnés vers Amazon Redshift sans serveur 2566
Mise à jour des requêtes tout en restant sur un cluster provisionné 2567
Amélioration du suivi des identificateurs de requêtes à l'aide des vues de surveillance SYS ... 2567
exemple 2568
Identifiants de requête, de processus et de session dans la table système 2575
Vues de métadonnées SVV 2575
SVV_ACTIVE_CURSORS 2578
SVV_ALL_COLUMNS 2579
SVV_ALL_SCHEMAS 2581
SVV_ALL_TABLES 2583
SVV_ALTER_TABLE_RECOMMENDATIONS 2584
SVV_ATTACHED_MASKING_POLICY 2586
SVV_COLUMNS 2589
SVV_COLUMN_PRIVILEGES 2591
SVV_COPY_JOB_INTEGRATIONS 2593
SVV_DATABASE_PRIVILEGES 2594
SVV_DATASHARE_PRIVILEGES 2596
SVV_DATASHARES 2597
SVV_DATASHARE_CONSUMERS 2601
SVV_DATASHARE_OBJECTS 2602
SVV_DEFAULT_PRIVILEGES 2604
SVV_DISKUSAGE 2606
SVV_EXTERNAL_COLUMNS 2609
SVV_EXTERNAL_DATABASES 2610
SVV_EXTERNAL_PARTITIONS 2611
SVV_EXTERNAL_SCHEMAS 2612
SVV_EXTERNAL_TABLES 2613
SVV_FUNCTION_PRIVILEGES 2615
SVV_GEOGRAPHY_COLUMNS 2617
SVV_GEOMETRY_COLUMNS 2618
SVV_IAM_PRIVILEGES 2619

SVV_IDENTITY_PROVIDERS 2621
SVV_INTEGRATION 2622
SVV_INTEGRATION_TABLE_MAPPAGE 2624
SVV_INTEGRATION_TABLE_STATE 2627
SVV_INTERLEAVED_COLUMNS 2629
SVV_LANGUAGE_PRIVILEGES 2630
SVV_MASKING_POLICY 2631
SVV_ML_MODEL_INFO 2632
SVV_ML_MODEL_PRIVILEGES 2634
SVV_MV_DEPENDENCY 2635
SVV_MV_INFO 2637
SVV_QUERY_INFLIGHT 2639
SVV_QUERY_STATE 2640
SVV_REDSHIFT_COLUMNS 2644
SVV_REDSHIFT_DATABASES 2647
SVV_REDSHIFT_FUNCTIONS 2649
SVV_REDSHIFT_SCHEMA_QUOTA 2650
SVV_REDSHIFT_SCHEMAS 2651
SVV_REDSHIFT_TABLES 2653
SVV_RELATION_PRIVILEGES 2654
SVV_RLS_APPLIED_POLICY 2656
SVV_RLS_ATTACHED_POLICY 2658
SVV_RLS_POLICY 2659
SVV_RLS_RELATION 2660
SVV_ROLE_GRANTS 2662
SVV_ROLES 2663
SVV_SCHEMA_PRIVILEGES 2663
SVV_SCHEMA_QUOTA_STATE 2665
SVV_SYSTEM_PRIVILEGES 2667
SVV_TABLE_INFO 2668
SVV_TABLES 2673
SVV_TRANSACTIONS 2674
SVV_USER_GRANTS 2676
SVV_USER_INFO 2678
SVV_VACUUM_PROGRESS 2679
SVV_VACUUM_SUMMARY 2681

Vues de surveillance SYS 2684
SYS_ANALYZE_COMPRESSION_HISTORY 2685
SYS_ANALYZE_HISTORY 2688
SYS_APPLIED_MASKING_POLICY_LOG 2690
OPTIMISATION DE LA TABLE SYS_AUTOMATIQUE 2692
SYS_CHILD_QUERY_TEXT 2693
SYS_CONNECTION_LOG 2695
SYS_COPY_JOB 2699
SYS_COPY_JOB_DETAIL 2700
SYS_COPY_JOB_INFO 2702
SYS_COPY_REPLACEMENTS 2703
SYS_DATASHARE_CHANGE_LOG 2704
SYS_DATASHARE_CROSS_REGION_USAGE 2708
SYS_DATASHARE_USAGE_CONSUMER 2709
SYS_DATASHARE_USAGE_PRODUCER 2710
SYS_EXTERNAL_QUERY_DETAIL 2712
SYS_EXTERNAL_QUERY_ERROR 2715
SYS_INTEGRATION_ACTIVITY 2718
SYS_INTEGRATION_TABLE_ACTIVITY 2720
SYS_INTEGRATION_TABLE_STATE_CHANGE 2721
SYS_LOAD_DETAIL 2723
SYS_LOAD_ERROR_DETAIL 2726
SYS_LOAD_HISTORY 2729
SYS_MV_REFRESH_HISTORY 2733
SYS_MV_STATE 2736
SYS_PROCEDURE_CALL 2739
SYS_PROCEDURE_MESSAGES 2741
SYS_QUERY_DETAIL 2743
SYS_QUERY_EXPLAIN 2752
SYS_QUERY_HISTORY 2756
SYS_QUERY_TEXT 2769
SYS_RESTORE_LOG 2772
SYS_RESTORE_STATE 2775
SYS_SCHEMA_QUOTA_VIOLATIONS 2777
SYS_SERVERLESS_USAGE 2779
SYS_SESSION_HISTORY 2782

SYS_SPATIAL_SIMPLIFY 2783
SYS_STREAM_SCAN_ERRORS 2785
SYS_STREAM_SCAN_STATES 2786
SYS_TRANSACTION_HISTORY 2788
SYS_UDF_LOG 2791
SYS_UNLOAD_DETAIL 2793
SYS_UNLOAD_HISTORY 2795
SYS_USERLOG 2797
SYS_VACUUM_HISTORY 2799
Cartographie des vues système pour la migration vers les vues de surveillance SYS 2803
SYS_QUERY_HISTORY 2804
SYS_QUERY_DETAIL 2805
SYS_RESTORE_LOG 2806
SYS_RESTORE_STATE 2806
SYS_TRANSACTION_HISTORY 2807
SYS_QUERY_TEXT 2807
SYS_CONNECTION_LOG 2807
SYS_SESSION_HISTORY 2807
SYS_LOAD_DETAIL 2807
SYS_LOAD_HISTORY 2808
SYS_LOAD_ERROR_DETAIL 2808
SYS_UNLOAD_HISTORY 2808
SYS_UNLOAD_DETAIL 2808
SYS_COPY_REPLACEMENTS 2808
SYS_DATASHARE_USAGE_CONSUMER 2809
SYS_DATASHARE_USAGE_PRODUCER 2809
SYS_DATASHARE_CROSS_REGION_USAGE 2809
SYS_DATASHARE_CHANGE_LOG 2809
SYS_EXTERNAL_QUERY_DETAIL 2809
SYS_EXTERNAL_QUERY_ERROR 2810
SYS_VACUUM_HISTORY 2810
SYS_ANALYZE_HISTORY 2810
SYS_ANALYZE_COMPRESSION_HISTORY 2810
SYS_MV_REFRESH_HISTORY 2810
SYS_MV_STATE 2811
SYS_PROCEDURE_CALL 2811

SYS_PROCEDURE_MESSAGES ... 2811
SYS_UDF_LOG ... 2811
SYS_USERLOG ... 2811
SYS_SCHEMA_QUOTA_VIOLATIONS ... 2812
SYS_SPATIAL_SIMPLIFY ... 2812
Surveillance système (clusters mis en service uniquement) ... 2812
Vues STL pour la journalisation ... 2812
Tables STV pour les données d'instantanés ... 2960
Vues SVCS pour le cluster principal et les clusters de mise à l'échelle de la simultanéité ... 3020
Vues SVL pour le cluster principal ... 3051
Tables catalogue système ... 3131
PG_ATTRIBUTE_INFO ... 3132
PG_CLASS_INFO ... 3132
PG_DATABASE_INFO ... 3134
PG_DEFAULT_ACL ... 3135
PG_EXTERNAL_SCHEMA ... 3139
PG_LIBRARY ... 3140
PG_PROC_INFO ... 3141
PG_STATISTIC_INDICATOR ... 3142
PG_TABLE_DEF ... 3143
PG_USER_INFO ... 3145
Interrogation des tables catalogue ... 3146
Référence de configuration ... 3153
Modification de la configuration du serveur ... 3154
analyze_threshold_percent ... 3155
Valeurs (par défaut en gras) ... 3155
Description ... 3155
Exemples ... 3156
cast_super_null_on_error ... 3156
Valeurs (par défaut en gras) ... 3156
Description ... 3156
datashare_break_glass_session_var ... 3156
Valeurs (par défaut en gras) ... 3156
Description ... 3156
exemple ... 3157
datestyle ... 3157

Valeurs (par défaut en gras) 3157
Description 3156
exemple 3158
default_geometry_encoding 3158
Valeurs (par défaut en gras) 3158
Description 3156
describe_field_name_in_uppercase 3158
Valeurs (par défaut en gras) 3158
Description 3156
exemple 3158
downcase_delimited_identificateur 3159
Valeurs (par défaut en gras) 3159
Description 3156
Notes d'utilisation 3159
enable_case_sensitive_identifier 3160
Valeurs (par défaut en gras) 3160
Description 3161
Exemples 3161
Notes d'utilisation 3162
enable_case_sensitive_super_attribute 3164
Valeurs (par défaut en gras) 3164
Description 3164
Exemples 3165
Notes d'utilisation 3166
enable_numeric_rounding 3166
Valeurs (par défaut en gras) 3166
Description 3166
exemple 3167
enable_result_cache_for_session 3168
Valeurs (par défaut en gras) 3168
Description 3168
exemple 3168
enable_vacuum_boost 3169
Valeurs (par défaut en gras) 3169
Description 3156
error_on_nondeterministic_update 3169

Valeurs (par défaut en gras) 3169
Description 3156
exemple 3158
extra_float_digits 3170
Valeurs (par défaut en gras) 3170
Description 3170
exemple 3170
interval_forbid_composite_literals 3171
Valeurs (par défaut en gras) 3171
Description 3156
json_serialization_enable 3172
Valeurs (par défaut en gras) 3172
Description 3156
json_serialization_parse_nested_strings 3172
Valeurs (par défaut en gras) 3172
Description 3156
max_concurrency_scaling_clusters 3173
Valeurs (par défaut en gras) 3173
Description 3173
max_cursor_result_set_size 3173
Valeurs (par défaut en gras) 3173
Description 3173
mv_enable_aqmv_for_session 3173
Valeurs (par défaut en gras) 3173
Description 3173
navigate_super_null_on_error 3174
Valeurs (par défaut en gras) 3174
Description 3156
parse_super_null_on_error 3174
Valeurs (par défaut en gras) 3174
Description 3156
pg_federation_repeatable_read 3174
Valeurs (par défaut en gras) 3174
Description 3156
Exemples 3175
query_group 3175

Valeurs (par défaut en gras) 3175
Description 3175
search_path 3176
Valeurs (par défaut en gras) 3176
Description 3176
exemple 3177
spectrum_enable_pseudo_columns 3178
Valeurs (par défaut en gras) 3178
Description 3178
exemple 3178
enable_spectrum_oid 3179
Valeurs (par défaut en gras) 3179
Description 3179
exemple 3179
spectrum_query_maxerror 3179
Valeurs (par défaut en gras) 3179
Description 3179
exemple 3179
statement_timeout 3180
Valeurs (par défaut en gras) 3180
Description 3180
exemple 3180
stored_proc_log_min_messages 3181
Valeurs (par défaut en gras) 3181
Description 3156
timezone 3181
Valeurs (par défaut en gras) 3181
Syntaxe 3181
Description 3182
Formats de fuseau horaire 3182
Exemples 3184
use_fips_ssl 3185
Valeurs (par défaut en gras) 3185
Description 3156
wlm_query_slot_count 3186
Valeurs (par défaut en gras) 3186

Description 3186
Exemples 3187
Historique de la documentation 3188
Mises à jour antérieures 3200
..... mmmccxxxv

Présentation d'Amazon Redshift

Bienvenue dans le guide du développeur de bases de données Amazon Redshift. Ce guide vise à vous aider à comprendre comment utiliser Amazon Redshift pour créer et gérer un entrepôt de données. Si vous travaillez avec des bases de données en tant que concepteur, développeur de logiciels ou administrateur, ce guide vous fournit les informations dont vous avez besoin pour concevoir, créer, interroger et gérer votre entrepôt de données.

Amazon Redshift est un service d'entrepôt des données entièrement géré dans le cloud. Amazon Redshift sans serveur vous permet d'accéder aux données et de les analyser sans les configurations habituelles d'un entrepôt des données provisionné. Les ressources sont automatiquement provisionnées et la capacité de l'entrepôt des données est intelligemment mise à l'échelle afin d'offrir des performances rapides, même pour les charges de travail les plus exigeantes et les plus imprévisibles. Vous ne payez pas de frais lorsque l'entrepôt des données est inactif, vous ne payez donc que ce que vous utilisez. Quelle que soit la taille du jeu de données, vous pouvez charger les données et commencer à effectuer des requêtes immédiatement dans l'éditeur de requête Amazon Redshift v2 ou dans votre outil d'informatique décisionnelle (BI) préféré. Profitez du meilleur rapport prix/performances et de fonctionnalités SQL familières dans un easy-to-use environnement sans administration.

Rubriques

- Conditions préalables à l'utilisation d'Amazon Redshift
- Architecture d'Amazon Redshift
- Exemple de base de données

Conditions préalables à l'utilisation d'Amazon Redshift

Cette rubrique décrit les conditions requises pour utiliser Amazon Redshift.

Avant d'utiliser ce guide, vous devriez lire Commencer avec les entrepôts de données Redshift Serverless, qui explique comment effectuer les tâches suivantes.

- Créez un entrepôt des données avec Amazon Redshift sans serveur.
- Chargement d'un exemple de données avec l'éditeur de requête d'Amazon Redshift v2
- Chargement de données depuis Amazon S3.

Vous devez également savoir comment utiliser votre client SQL et avoir une connaissance fondamentale du langage SQL.

Architecture d'Amazon Redshift

Cette rubrique vous aide à comprendre les composants qui constituent Amazon Redshift.

Un entrepôt des données Amazon Redshift est un système de gestion et de requête de base de données relationnelle de niveau entreprise.

Amazon Redshift prend en charge les connexions client avec de nombreux types d'applications, notamment les outils de business intelligence (BI), de reporting, de données et d'analytique.

Lorsque vous exécutez des requêtes analytiques, vous extrayez, comparez et évaluez de grandes quantités de données dans le cadre d'opérations à plusieurs étapes, afin d'obtenir un résultat final.

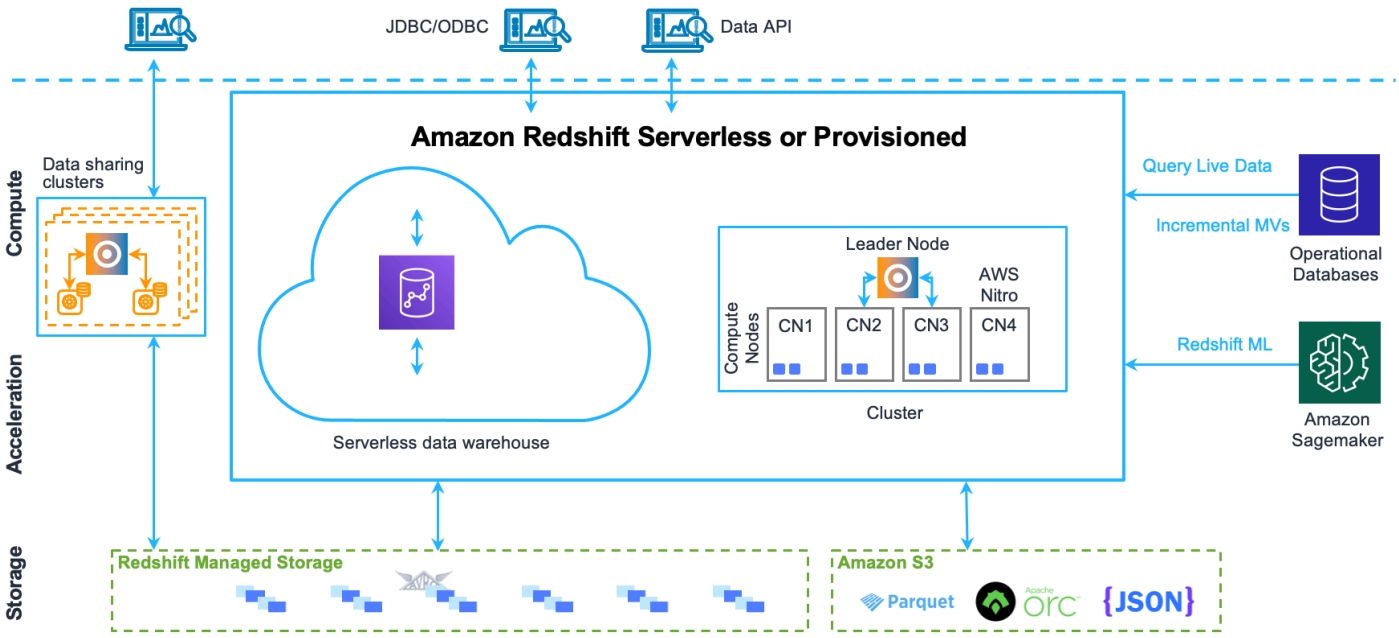
Amazon Redshift assure un stockage efficace et des performances de requête optimales en combinant un traitement massivement parallèle, un stockage de données en colonnes et des schémas d'encodage de compression de données ciblés et très efficaces. Cette section présente une introduction à l'architecture système d'Amazon Redshift.

Rubriques

- Architecture système de l'entrepôt des données
- Performances d'Amazon Redshift
- Stockage en colonnes
- Gestion de la charge de travail
- Utilisation d'Amazon Redshift avec d'autres services

Architecture système de l'entrepôt des données

Cette section explique les composants qui constituent l'architecture de l'entrepôt de données Amazon Redshift, comme le montre la figure suivante.



Applications clientes

Amazon Redshift s'intègre à divers outils de chargement de données et d'ETL (extraction, transformation et chargement), ainsi qu'à des outils de reporting, d'exploration de données et d'analyse de la Business Intelligence (BI). Amazon Redshift est basé sur PostgreSQL en norme ouverte, de sorte que la plupart des applications clientes SQL existantes fonctionneront avec des changements minimes. Pour plus d'informations sur les différences importantes entre Amazon Redshift SQL et PostgreSQL, consultez Amazon Redshift et PostgreSQL.

Clusters

Le composant principal de l'infrastructure d'un entrepôt des données Amazon Redshift est un cluster.

Un cluster est composé d'un ou plusieurs nœuds de calcul. Si un cluster est provisionné avec deux nœuds de calcul ou plus, un nœud principal supplémentaire coordonne les nœuds de calcul et gère la communication externe. Votre application cliente n'interagit directement qu'avec le nœud principal. Les nœuds de calcul sont transparents pour les applications externes.

Nœud principal

Le nœud principal gère les communications avec les programmes clients et toute la communication avec les nœuds de calcul. Il analyse et développe des plans d'exécution pour effectuer des opérations de base de données : en particulier, la série d'étapes nécessaires pour obtenir des résultats pour les requêtes complexes. D'après le plan d'exécution, le nœud principal compile le

code, distribue le code compilé aux nœuds de calcul et attribue une partie des données à chaque nœud de calcul.

Le nœud principal distribue les instructions SQL aux nœuds de calcul uniquement quand une requête fait référence aux tables stockées sur les nœuds de calcul. Toutes les autres requêtes s'exécutent exclusivement sur le nœud principal. Amazon Redshift est conçu pour implémenter certaines fonctions SQL uniquement sur le nœud principal. Une requête qui utilise une de ces fonctions retourne une erreur si elle fait référence aux tables qui résident sur les nœuds de calcul. Pour plus d'informations, consultez Fonctions SQL prises en charge sur le nœud principal.

Nœuds de calcul

Le nœud principal compile le code des éléments du plan d'exécution et affecte le code aux nœuds de calcul. Les nœuds de calcul exécutent le code compilé et renvoient les résultats intermédiaires au nœud principal pour l'agrégation finale.

Chaque nœud de calcul a ses propres UC et mémoire dédiés, déterminées par le type de nœud. Lorsque votre charge de travail augmente, vous pouvez augmenter la capacité de calcul d'un cluster en augmentant le nombre de nœuds, en mettant à niveau le type de nœud, ou les deux.

Amazon Redshift propose plusieurs types de nœuds pour vos besoins de calcul. Pour plus de détails sur chaque type de nœud, consultez Cluster Amazon Redshift dans le Guide de la gestion du cluster Amazon Redshift.

Redshift Managed Storage

Les données de l'entrepôt des données sont stockées dans un niveau stockage séparé Redshift Managed Storage (RMS). RMS permet d'augmenter votre stockage à plusieurs pétaoctets à l'aide du stockage Amazon S3. RMS vous permet de faire mettre à l'échelle et de payer le calcul et le stockage de manière indépendante, de sorte que vous pouvez dimensionner votre cluster uniquement en fonction de vos besoins en matière de calcul. Il utilise automatiquement le stockage local à haute performance sur SSD comme cache de niveau 1. Il tire également parti d'optimisations telles que la température des blocs de données, leur âge et les modèles de charge de travail, pour fournir des performances élevées tout en adaptant automatiquement le stockage vers Amazon S3 en cas de besoin, sans qu'aucune action ne soit requise.

Tranches de nœud

Un nœud de calcul est divisé en tranches. Chaque tranche se voit attribuer une partie de la mémoire et de l'espace disque du nœud, où elle traite une partie de la charge de travail affectée au nœud.

Le nœud principal gère la distribution des données aux tranches et attribue la charge de travail des requêtes ou autres opérations de base de données aux tranches. Les tranches travaillent alors en parallèle pour terminer l'opération.

Le nombre de tranches par nœud est déterminé par la taille de nœud du cluster. Pour plus d'informations sur le nombre de tranches pour chaque taille de nœud, consultez À propos des clusters et des nœuds dans le Guide de gestion Amazon Redshift.

Lorsque vous créez une table, vous pouvez éventuellement spécifier une colonne comme clé de distribution. Lorsque la table est chargée avec les données, les lignes sont distribuées aux tranches des nœuds selon la clé de distribution définie pour une table. Le choix d'une bonne clé de distribution permet à Amazon Redshift d'utiliser le traitement parallèle pour charger les données et exécuter les requêtes efficacement. Pour plus d'informations sur le choix d'une clé de distribution, consultez Choisir le meilleur style de distribution.

Réseau interne

Amazon Redshift tire parti de connexions à large bande passante, de la proximité et de protocoles de communication personnalisés pour fournir une communication réseau privée à très haut débit entre le nœud principal et les nœuds de calcul. Les nœuds de calcul s'exécutent sur un réseau isolé distinct auquel les applications clientes n'accèdent jamais directement.

Bases de données

Un cluster contient une ou plusieurs bases de données. Les données utilisateur sont stockées sur les nœuds de calcul. Votre client SQL communique avec le nœud principal, qui à son tour coordonne l'exécution de la requête avec les nœuds de calcul.

Amazon Redshift est un système de gestion de base de données relationnelle (SGBDR), compatible de ce fait avec d'autres applications SGBDR. Même s'il fournit les mêmes fonctionnalités qu'un SGBDR classique, y compris les fonctions de traitement transactionnel en ligne (OLTP) comme l'insertion et la suppression de données, Amazon Redshift est optimisé pour l'analyse hautes performances et la création de rapports de jeux de données très volumineux.

Amazon Redshift est basé sur PostgreSQL. Amazon Redshift et PostgreSQL présentent un certain nombre de différences très importantes que vous devez prendre en compte lors de la conception et du développement de vos applications d'entrepôt des données. Pour plus d'informations sur les différences entre Amazon Redshift SQL et PostgreSQL, voir Amazon Redshift et PostgreSQL.

Performances d’Amazon Redshift

Cette rubrique décrit les composants Amazon Redshift qui améliorent les performances. La compréhension de ces composants vous aidera à optimiser les performances et à résoudre les problèmes de performances médiocres avec Amazon Redshift.

Amazon Redshift assure une exécution extrêmement rapide des requêtes en utilisant ces fonctions de performance.

Rubriques

- Traitement massivement parallèle
- Stockage des données en colonnes
- Compression des données
- Optimiseur de requête
- Mise en cache du résultat
- Code compilé

Traitement massivement parallèle

Le traitement massivement parallèle (MPP) permet l’exécution rapide des requêtes les plus complexes œuvrant sur de grandes quantités de données. Plusieurs nœuds de calcul gèrent tout le traitement des requêtes conduisant à l’agrégation du résultat final, avec chaque cœur de chaque nœud exécutant les mêmes segments de requêtes compilés sur des parties de l’ensemble des données.

Amazon Redshift distribue les lignes d’une table aux différents nœuds de calcul afin que les données puissent être traitées en parallèle. En sélectionnant une clé de distribution appropriée pour chaque table, vous pouvez optimiser la distribution des données pour équilibrer l’application et réduire le mouvement de données d’un nœud à un autre. Pour plus d'informations, consultez Choisir le meilleur style de distribution.

Le chargement de données à partir de fichiers plats tire parti du traitement en répartissant la charge de travail entre plusieurs nœuds, tout en assurant des lectures simultanées à partir de plusieurs fichiers. Pour plus d’informations sur le chargement des données dans les tables, consultez Bonnes pratiques de chargement des données sur Amazon Redshift.

Stockage des données en colonnes

Le stockage en colonnes des tables de base de données réduit considérablement les exigences globales d'I/O disque et constitue un facteur important dans l'optimisation des performances des requêtes analytiques. Pour de plus amples informations, veuillez consulter Stockage en colonnes.

Lorsque les colonnes sont triées de manière adéquate, le processeur de requêtes est capable de filtrer rapidement un large sous-ensemble de blocs de données. Pour plus d'informations, consultez Choix de la meilleure clé de tri.

Compression des données

La compression des données réduit les besoins en stockage et, de ce fait, les I/O disque, ce qui améliore les performances des requêtes. Lorsque vous exécutez une requête, les données compressées sont lues en mémoire, puis décompressées au cours de l'exécution des requêtes. Le fait de charger moins de données en mémoire permet à Amazon Redshift d'allouer plus de mémoire à l'analyse des données. Comme le stockage en colonnes stocke des données similaires de manière séquentielle, Amazon Redshift est en mesure d'appliquer des encodages de compression adaptatifs spécifiquement liés aux types de données en colonnes. Le meilleur moyen d'activer la compression des données sur les colonnes de la table est de permettre à Amazon Redshift d'appliquer des encodages de compression optimaux lorsque vous chargez la table avec des données. Pour en savoir plus sur l'utilisation de la compression automatique des données, consultez Chargement des tables avec compression automatique.

Optimiseur de requête

Le moteur d'exécution des requêtes Amazon Redshift intègre un optimiseur de requête qui prend en charge le traitement hautement parallèle et tire aussi parti du stockage de données orienté colonnes. L'optimiseur de requête Amazon Redshift met en œuvre des améliorations et des extensions importantes pour le traitement des requêtes analytiques complexes qui incluent souvent les jointures de plusieurs tables, les sous-requêtes et l'agrégation. Pour en savoir plus sur l'optimisation des requêtes, consultez Réglage des performances des requêtes.

Mise en cache du résultat

Pour raccourcir le temps d'exécution des requêtes et améliorer les performances du système, Amazon Redshift met en cache le résultat de certains types de requêtes dans la mémoire du nœud principal. Quand un utilisateur envoie une requête, Amazon Redshift recherche une copie valide du résultat de la requête dans le cache. Si une correspondance est trouvée dans le cache des résultats,

Amazon Redshift utilise les résultats mis en cache et n'exécute pas la requête. La mise en cache du résultat est totalement transparente pour l'utilisateur.

La mise en cache du résultat est activée par défaut. Pour désactiver la mise en cache du résultat sur la séance en cours, définissez le paramètre de configuration enable_result_cache_for_session sur `off`.

Amazon Redshift utilise les résultats mis en cache quand il reçoit une nouvelle requête si l'ensemble des conditions suivantes sont satisfaites :

- L'utilisateur qui soumet la requête a des autorisations d'accès sur les objets concernés par la requête.
- La table ou les vues incluses dans la requête n'ont pas été modifiées.
- La requête n'utilise pas une fonction qui nécessite une évaluation à chaque exécution (ex. : GETDATE).
- La requête ne fait pas référence à des tables externes Amazon Redshift Spectrum.
- Les paramètres de configuration qui pourraient affecter le résultat de la requête n'ont pas été modifiés.
- La syntaxe de la requête correspond à celle mise en cache.

Pour maximiser l'efficacité du cache et l'utilisation efficace des ressources, Amazon Redshift ne met pas en cache certains grands ensembles de résultats de requêtes. Amazon Redshift détermine s'il faut mettre en cache les résultats des requêtes en fonction d'un certain nombre de facteurs. Le nombre d'entrées en cache et le type d'instance de votre cluster Amazon Redshift entrent notamment en jeu.

Pour savoir si une requête exploite le cache de résultats, interrogez la vue système SVL_QLOG. Si la requête utilise le cache de résultats, la colonne source_query indique l'ID de la requête source. Si la mise en cache n'a pas été utilisée, la valeur indiquée est NULL.

L'exemple suivant montre que les requêtes soumises par les ID utilisateur 104 et 102 exploitent les résultats en cache des requêtes exécutées par l'ID utilisateur 100.

```
select userid, query, elapsed, source_query from svl_qlog
where userid > 1
order by query desc;

userid | query  | elapsed  | source_query
```

```
-------+--------+----------+-------------
   104 | 629035 |       27 |       628919
   104 | 629034 |       60 |       628900
   104 | 629033 |       23 |       628891
   102 | 629017 |  1229393 |
   102 | 628942 |       28 |       628919
   102 | 628941 |       57 |       628900
   102 | 628940 |       26 |       628891
   100 | 628919 | 84295686 |
   100 | 628900 | 87015637 |
   100 | 628891 | 58808694 |
```

Code compilé

Le nœud principal répartit le code compilé entièrement optimisé entre tous les nœuds d'un cluster. La compilation de la requête réduit le traitement associé à un interpréteur et, par conséquent, augmente la vitesse d'exécution, en particulier pour les requêtes complexes. Le code compilé est mis en cache et partagé entre les séances du même cluster. Par conséquent, les exécutions futures de la même requête seront plus rapides, souvent même avec des paramètres différents.

Le moteur d'exécution de requête compile un code différent pour les protocoles de connexion JDBC et ODBC, si bien que deux clients utilisant chacun un protocole différent supportent le coût initial de compilation du code. En revanche, les clients qui utilisent le même protocole bénéficient du partage du code mis en cache.

Stockage en colonnes

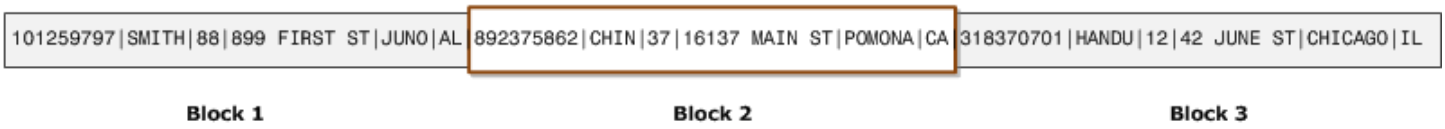
Cette section décrit le stockage en colonnes, qui est la méthode utilisée par Amazon Redshift pour stocker efficacement les données tabulaires.

Le stockage en colonnes des tables de base de données constitue un facteur important dans l'optimisation des performances des requêtes analytiques car il réduit considérablement les exigences globales d'I/O disque. Il permet de réduire la quantité de données que vous devez charger sur le disque.

La série d'illustrations suivantes décrit comment le stockage de données en colonnes garantit l'efficacité et comment celle-ci se traduit par une extraction efficace des données en mémoire.

Cette première illustration montre comment les enregistrements des tables de base de données sont généralement stockés en blocs de disque par ligne.

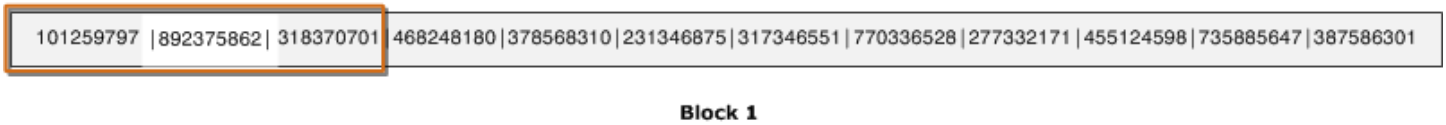
SSN	Name	Age	Addr	City	St
101259797	SMITH	88	899 FIRST ST	JUNO	AL
892375862	CHIN	37	16137 MAIN ST	POMONA	CA
318370701	HANDU	12	42 JUNE ST	CHICAGO	IL



Dans une table de base de données relationnelle classique, chaque ligne contient les valeurs de champ d'un seul enregistrement. Dans le stockage de base de données en lignes, les blocs de données stockent les valeurs de manière séquentielle pour chaque colonne composant la totalité de la ligne. Si la taille de bloc est inférieure à celle de la taille d'un enregistrement, le stockage d'un enregistrement complet peut nécessiter plus d'un bloc. Si la taille de bloc est supérieure à celle de la taille d'un enregistrement, le stockage d'un enregistrement complet peut nécessiter moins d'un bloc, ce qui entraîne une utilisation inefficace de l'espace disque. Dans les applications de traitement transactionnel en ligne (OLTP), la plupart des transactions impliquent une lecture et une écriture fréquentes de toutes les valeurs de l'ensemble des enregistrements, généralement un enregistrement ou un petit nombre d'enregistrements à la fois. En conséquence, le stockage en ligne est optimal pour les bases de données OLTP.

L'illustration suivante montre comment, grâce au stockage en colonnes, les valeurs de chaque colonne sont stockées de manière séquentielle en blocs de disque.

SSN	Name	Age	Addr	City	St
101259797	SMITH	88	899 FIRST ST	JUNO	AL
892375862	CHIN	37	16137 MAIN ST	POMONA	CA
318370701	HANDU	12	42 JUNE ST	CHICAGO	IL



A l'aide du stockage en colonnes, chaque bloc de données stocke les valeurs d'une seule colonne pour plusieurs lignes. Au fur et à mesure que les enregistrements entrent dans le système, Amazon Redshift convertit les données de façon transparente en stockage en colonnes pour chacune des colonnes.

Dans cet exemple simplifié, à l'aide du stockage en colonnes, chaque bloc de données contient les valeurs des champs de colonne pour trois fois plus d'enregistrements que le stockage en lignes. Cela signifie que la lecture du même nombre de valeurs de champs de colonne pour le même nombre d'enregistrements nécessite un tiers des opérations d'I/O par rapport au stockage en lignes. En pratique, si vous utilisez des tables avec des nombres très élevés de lignes et de colonnes, l'efficacité du stockage est encore plus grande.

Un avantage supplémentaire est que, dans la mesure où chaque bloc contient le même type de données, les données de bloc peuvent utiliser un système de compression sélectionné spécifiquement pour le type de données de la colonne, réduisant encore plus l'espace disque et les I/O. Pour en savoir plus sur les encodages de compression basés sur les types de données, consultez encodages de compression.

Les économies d'espace obtenues pour stocker les données sur disque se répercutent dans l'extraction et dans le stockage de ces données en mémoire. Comme la plupart des opérations de base de données ont seulement besoin d'accéder à un petit nombre de colonnes à la fois ou de les exploiter, vous pouvez économiser de l'espace mémoire en extrayant uniquement les blocs des colonnes dont vous avez réellement besoin pour ne requête. Là où les transactions OLTP impliquent généralement la plupart ou la totalité des colonnes d'une ligne d'un petit nombre de documents, les requêtes sur les entrepôts des données ne lisent généralement que quelques colonnes d'un très grand nombre de lignes. Cela signifie que la lecture du même nombre de valeurs de champs de colonne pour le même nombre de ligne nécessite une fraction des opérations d'I/O. Elle utilise une fraction de la mémoire qui serait nécessaire pour traiter des blocs par ligne. En pratique, l'utilisation de tables avec de très grands nombres de colonnes et de lignes permet d'obtenir des gains en efficacité proportionnellement supérieurs. Par exemple, imaginons qu'une table contienne 100 colonnes. Une requête qui utilise cinq colonnes a uniquement besoin de lire environ 5 % des données contenues dans la table. Cette économie se reproduit pour les éventuels milliards, voire billions, d'enregistrements des bases de données volumineuses. En revanche, une base de données en lignes lit tout aussi bien les blocs contenant les 95 % de colonnes non nécessaires.

La taille d'un bloc de base de données classique varie de 2 à 32 Ko. Amazon Redshift utilise une taille de bloc de 1 Mo, ce qui est plus efficace et réduit le nombre de requêtes d'I/O nécessaires pour exécuter un chargement de base de données ou toute autre opération relevant de l'exécution de la requête.

Gestion de la charge de travail

Cette section décrit la gestion de la charge de travail (WLM), qui vous aide à comprendre comment Amazon Redshift prépare et exécute les requêtes.

La gestion des charges de travail Amazon Redshift (WLM) permet de définir des priorités de gestion flexibles au sein des charges de travail afin que les requêtes courtes et rapides ne restent pas bloquées dans des files d'attente derrière des requêtes de longue durée. Amazon Redshift crée des files d'attente de requêtes lors de l'exécution en fonction des classes de service, qui définissent les paramètres de configuration pour différents types de files d'attente, y compris les files d'attente du système interne et les files d'attente accessibles aux utilisateurs. Du point de vue de l'utilisateur, une classe de service accessible par l'utilisateur et une file d'attente sont fonctionnellement équivalentes. Pour des raisons de cohérence, cette documentation utilise le terme file d'attente pour désigner une classe de service accessible par l'utilisateur aussi bien qu'une file d'attente à l'exécution.

Redshift propose une gestion automatique de la charge de travail, appelée WLM automatique, qui est réglée pour gérer différentes charges de travail et constitue la valeur par défaut recommandée. Avec le WLM automatique, Redshift détermine l'utilisation des ressources à mesure que les requêtes arrivent et détermine de manière dynamique s'il convient de les exécuter sur le cluster principal, sur un cluster de dimensionnement de la simultanéité ou de les envoyer à une file d'attente. (Lorsque les requêtes sont mises en file d'attente, le WLM automatique donne la priorité aux requêtes de courte durée.) Le WLM automatique maximise le débit total et vous permet de maintenir des ressources d'entrepôt de données efficaces. Vous exécutez des charges de travail sans avoir à vous soucier de leur taille ou de leur planification. Le WLM automatique est le mode par défaut pour les clusters provisionnés. Pour plus d'informations, consultez Implémentation du WLM automatique.

> **Note**
>
> Les groupes de travail Amazon Redshift Serverless utilisent toujours le WLM automatique.

Lorsque de nombreuses requêtes ou des requêtes gourmandes en ressources sont exécutées, la gestion de la charge de travail peut être étendue à des ressources de calcul supplémentaires lorsque les charges de travail sont mises en file d'attente sur des ressources locales. La mise à l'échelle de la simultanéité avec WLM automatique garantit des performances constantes pour un nombre pratiquement illimité d'utilisateurs et de requêtes simultanés.

Les clusters provisionnés Redshift proposent un WLM manuel si vous avez besoin d'une optimisation manuelle précise. Ici, le client gère l'allocation des ressources, la simultanéité des requêtes et la

mise en file d'attente. Lorsqu'une requête est exécutée, WLM l'affecte à une file d'attente en fonction du groupe d'utilisateurs de l'utilisateur ou en faisant correspondre un groupe de requêtes répertorié dans la configuration de la file d'attente. Ceci est configuré avec une étiquette de groupe de requêtes définie par l'utilisateur. Pour plus d'informations, consultez Implémentation du manuel WLM.

Bien que le WLM manuel puisse être affiné au fil du temps pour s'adapter à vos modèles de charge de travail, dans la plupart des cas, nous déconseillons son utilisation car sa nature statique peut vous empêcher de vous adapter à l'évolution des charges de travail au cours d'une journée ou sur une période prolongée. Cela nécessite une surveillance accrue et un réglage continu. En outre, dans de nombreux cas, le WLM manuel n'utilise pas les ressources de calcul aussi efficacement que le WLM automatique, par exemple si les files d'attente sont définies manuellement pour limiter la mémoire qui leur est allouée.

Un indicateur important pour mesurer le succès de la configuration de la gestion de la charge de travail est le débit du système, c'est-à-dire le nombre de requêtes traitées avec succès. Le débit du système est mesuré en requêtes par seconde. Pour plus d'informations sur les métriques du système, consultez la section Surveillance des performances du cluster Amazon Redshift.

Le moyen le plus simple de gérer votre configuration WLM consiste à utiliser la console de gestion Amazon Redshift. Vous pouvez également utiliser l'interface de ligne de commande (CLI) Amazon Redshift ou l'API Amazon Redshift. Pour plus d'informations sur la mise en œuvre et l'utilisation de la gestion de la charge de travail, consultez la section Implémentation de la gestion

Utilisation d’Amazon Redshift avec d’autres services

Cette section décrit comment vous pouvez utiliser d'autres services comme sources et destinations pour les données Amazon Redshift.

Amazon Redshift s'intègre à d'autres AWS services pour vous permettre de déplacer, de transformer et de charger vos données rapidement et de manière fiable, à l'aide de fonctionnalités de sécurité des données.

S3

Amazon Simple Storage Service (Amazon S3) est un service web qui stocke les données dans le cloud. Amazon Redshift exploite le traitement parallèle pour lire et charger les données à partir de plusieurs fichiers de données stockés dans les compartiments Amazon S3. Pour plus d'informations, consultez Chargement des données à partir d’Amazon S3.

Vous pouvez également utiliser le traitement parallèle pour exporter les données depuis votre entrepôt des données Amazon Redshift vers plusieurs fichiers de données sur Amazon S3. Pour de plus amples informations, veuillez consulter Déchargement de données dans Amazon Redshift.

DynamoDB

Amazon DynamoDB est un service de base de données NoSQL entièrement géré. Vous pouvez utiliser la commande COPY pour charger une table Amazon Redshift avec les données d'une seule table Amazon DynamoDB. Pour de plus amples informations, veuillez consulter Chargement de données à partir d'une table Amazon DynamoDB.

SSH

Vous pouvez utiliser la commande COPY dans Amazon Redshift pour charger des données depuis un ou plusieurs hôtes distants, tels que des clusters Amazon EMR, des EC2 instances Amazon ou d'autres ordinateurs. COPY se connecte aux hôtes distants à l'aide de SSH et exécute les commandes sur les hôtes distants pour générer les données. Amazon Redshift prend en charge plusieurs connexions simultanées. La commande COPY lit et charge la sortie de plusieurs hôtes source en parallèle. Pour de plus amples informations, veuillez consulter Chargement des données à partir des hôtes distants.

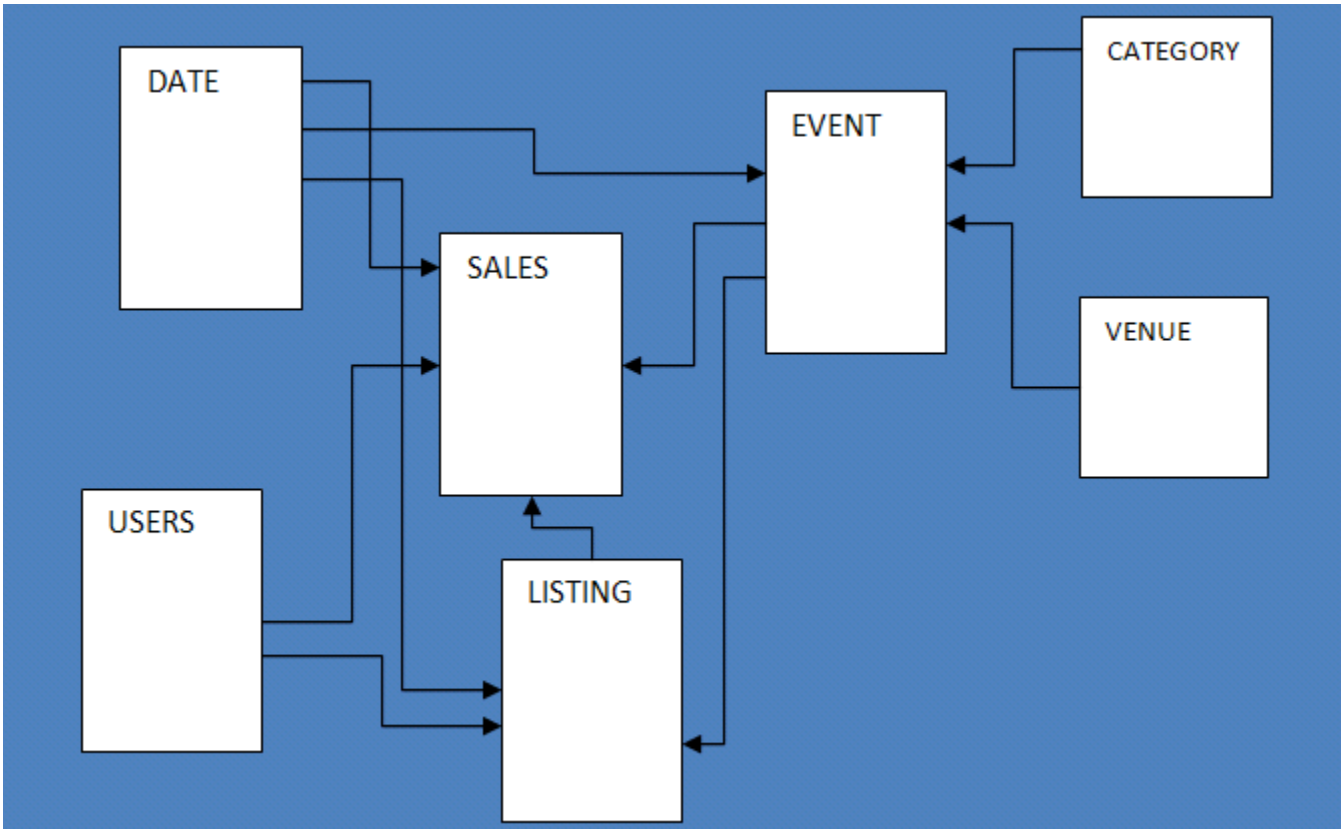
AWS DMS

Vous pouvez migrer des données vers Amazon Redshift à l'aide de. AWS Database Migration Service AWS DMS peut migrer vos données vers et depuis les bases de données commerciales et open source les plus utilisées, telles qu'Oracle, PostgreSQL, Microsoft SQL Server, Amazon Redshift, le cluster de base de données Aurora, DynamoDB, Amazon S3, MariaDB et MySQL. Pour plus d'informations, consultez Utilisation d'une base de données Amazon Redshift comme cible pour AWS Database Migration Service.

Exemple de base de données

Cette section décrit TICKIT, un exemple de base de données utilisé dans les exemples de documentation Amazon Redshift.

Cette petite base de données se compose de sept tables : deux tables de faits et cinq dimensions. Vous pouvez charger le jeu de données TICKIT en suivant les étapes de l'étape 4 : Charger les données d'Amazon S3 vers Amazon Redshift dans le guide de démarrage Amazon Redshift.



Cet exemple d'application de base de données aide les analystes à suivre l'activité de ventes du site web fictif TICKIT, où les utilisateurs achètent et vendent en ligne des billets pour des événements sportifs, des spectacles et des concerts. En particulier, les analystes peuvent identifier le mouvement des billets au fil du temps, les taux de réussite des vendeurs et les événements, lieux et saisons qui obtiennent les meilleurs résultats de vente. Les analystes peuvent utiliser ces informations pour offrir des primes aux acheteurs et aux vendeurs qui fréquentent le site, pour attirer de nouveaux utilisateurs et pour attirer promotions et publicités.

Par exemple, la requête suivante recherche les cinq meilleurs vendeurs à San Diego, en fonction du nombre de billets vendus en 2008 :

```
select sellerid, username, (firstname ||' '|| lastname) as name,
city, sum(qtysold)
from sales, date, users
where sales.sellerid = users.userid
and sales.dateid = date.dateid
and year = 2008
and city = 'San Diego'
group by sellerid, username, name, city
order by 5 desc
```

```
limit 5;

sellerid | username |       name        |   city    | sum
---------+----------+-------------------+-----------+-----
49977 | JJK84WTE | Julie Hanson      | San Diego |  22
19750 | AAS23BDR | Charity Zimmerman | San Diego |  21
29069 | SVL81MEQ | Axel Grant        | San Diego |  17
43632 | VAG08HKW | Griffin Dodson    | San Diego |  16
36712 | RXT40MKU | Hiram Turner      | San Diego |  14
(5 rows)
```

La base de données utilisée pour les exemples de ce guide contient un ensemble réduit de données ; les deux tables de faits contiennent chacune moins de 200 000 lignes et les dimensions s'étendent de 11 lignes dans la table CATEGORY jusqu'à environ 50 000 lignes dans la table USERS.

En particulier, les exemples de base de données de ce guide illustrent les fonctions clés de la conception de tables Amazon Redshift :

- Distribution de données
- Tri de données
- Compression en colonnes

Pour plus d'informations sur les schémas des tables de la base de données TICKIT, choisissez les onglets suivants :

CATEGORY table

Nom de la colonne	Type de données	Description
CATID	SMALLINT	Clé primaire, valeur d'ID unique pour chaque ligne. Chaque ligne représente un type spécifique d'événement pour lequel les billets sont achetés et vendus.
CATGROUP	VARCHAR(10)	Nom descriptif d'un groupe d'événements, tel que **`Shows`** et **`Sports`**.
CATNAME	VARCHAR(10)	Nom descriptif bref d'un type d'événement au sein d'un groupe, tel que **`Opera`** et **`Musicals`**.

Nom de la colonne	Type de données	Description
CATDESC	VARCHAR(50)	Nom descriptif plus long du type d'événement, tel que **`Musical theatre`**.

DATE table

Nom de la colonne	Type de données	Description
DATEID	SMALLINT	Clé primaire, valeur d'ID unique pour chaque ligne. Chaque ligne correspond à un jour de l'année calendaire.
CALDATE	DATE	Date calendaire, telle que **`2008-06-24`** .
DAY	CHAR(3)	Jour de semaine (forme courte), tel que **`SA`**.
WEEK	SMALLINT	Numéro de semaine, tel que **`26`**.
MONTH	CHAR(5)	Nom du mois (forme courte), tel que **`JUN`**.
QTR	CHAR(5)	Numéro de trimestre (**`1`** à **`4`**).
YEAR	SMALLINT	Année sur quatre chiffres (**`2008`**).
HOLIDAY	BOOLEAN	Indicateur qui indique si le jour est un jour férié (USA).

EVENT table

Nom de la colonne	Type de données	Description
EVENTID	INTEGER	Clé primaire, valeur d'ID unique pour chaque ligne. Chaque ligne représente un événement distinct qui a lieu dans un lieu donné à un moment donné.

Nom de la colonne	Type de données	Description
VENUEID	SMALLINT	Référence de clé étrangère à la table VENUE.
CATID	SMALLINT	Référence de clé étrangère à la table CATEGORY.
DATEID	SMALLINT	Référence de clé étrangère à la table DATE.
EVENTNAME	VARCHAR(200)	Nom de l'événement, tel que **`Hamlet`** ou **`La Traviata`**.
STARTTIME	TIMESTAMP	Date complète et heure de début de l'événement, telles que **`2008-10-10 19:30:00`** .

VENUE table

Nom de la colonne	Type de données	Description
VENUEID	SMALLINT	Clé primaire, valeur d'ID unique pour chaque ligne. Chaque ligne représente un lieu spécifique où les événements ont lieu.
VENUENAME	VARCHAR(100)	Nom exact du lieu, tel que **`Cleveland Browns Stadium`**.
VENUECITY	VARCHAR(30)	Nom de la ville, tel que **`Cleveland`** .
VENUESTATE	CHAR(2)	État ou province abrégé sur deux lettres (États-Unis et Canada), tel que **`OH`**.
VENUESEATS	INTEGER	Nombre maximal de places disponibles sur le site, s'il est connu, tel que **`73200`**. A des fins de démonstration, cette colonne contient certaines valeurs null ou égales à zéro (0).

USERS table

Nom de la colonne	Type de données	Description
USERID	INTEGER	Clé primaire, valeur d'ID unique pour chaque ligne. Chaque ligne représente un utilisateur enregistré (un acheteur ou un vendeur, ou les deux) qui a proposé ou acheté des billets pour au moins un événement.
USERNAME	CHAR(8)	Nom d'utilisateur alphanumérique de 8 caractères, tel que **`PGL08LJI`**.
FIRSTNAME	VARCHAR(30)	Prénom de l'utilisateur, tel que **`Victor`**.
LASTNAME	VARCHAR(30)	Nom de l'utilisateur, tel que **`Hernandez`** .
CITY	VARCHAR(30)	Ville de l'utilisateur, telle que **`Naperville`** .
STATE	CHAR(2)	État de l'utilisateur, tel que **`GA`**.
EMAIL	VARCHAR(100)	Adresse électronique de l'utilisateur ; cette colonne contient des valeurs latines aléatoires, telles que **`turpis@accumsanlaoreet.org`** .
PHONE	CHAR(14)	Numéro de téléphone sur 14 caractères de l'utilisateur, tel que **`(818) 765-4255`**.
LIKESPORTS, ...	BOOLEAN	Série de 10 colonnes différentes qui identifient ce que l'utilisateur aime et n'aime pas à l'aide des valeurs **`true`** et **`false`**.

LISTING table

Nom de la colonne	Type de données	Description
LISTID	INTEGER	Clé primaire, valeur d'ID unique pour chaque ligne. Chaque ligne représente une liste d'un lot de places pour un événement spécifique.
SELLERID	INTEGER	Référence de clé étrangère à la table USERS, identifiant l'utilisateur qui vend les billets.
EVENTID	INTEGER	Référence de clé étrangère à la table EVENT.
DATEID	SMALLINT	Référence de clé étrangère à la table DATE.
NUMTICKETS	SMALLINT	Nombre de billets disponibles à la vente, tel que **2** ou **20**.
PRICEPERTICKET	DECIMAL(8,2)	Prix fixe d'un billet individuel, tel que **27.00** ou **206.00**.
TOTALPRICE	DECIMAL(8,2)	Prix total de cette liste (NUMTICKETS*PRICEPERTICKET).
LISTTIME	TIMESTAMP	Date complète et heure auxquelles la liste a été publiée, telles que **2008-03-18 07:19:35** .

SALES table

Nom de la colonne	Type de données	Description
SALESID	INTEGER	Clé primaire, valeur d'ID unique pour chaque ligne. Chaque ligne représente une vente d'une ou de plusieurs places pour un événement spécifique, comme proposé dans une liste spécifique.

Nom de la colonne	Type de données	Description
LISTID	INTEGER	Référence de clé étrangère à la table LISTING.
SELLERID	INTEGER	Référence de clé étrangère à la table USERS (l'utilisateur qui a vendu les billets).
BUYERID	INTEGER	Référence de clé étrangère à la table USERS (l'utilisateur qui a acheté les billets).
EVENTID	INTEGER	Référence de clé étrangère à la table EVENT.
DATEID	SMALLINT	Référence de clé étrangère à la table DATE.
QTYSOLD	SMALLINT	Nombre de billets vendus, de **1** à **8**. (Un maximum de 8 billets peut être vendu dans une même transaction).
PRICEPAID	DECIMAL(8,2)	Prix total payé pour les billets, tel que **75.00** ou **488.00**. Le prix individuel d'un billet est PRICEPAID/QTYSOLD.
COMMISSION	DECIMAL(8,2)	Commission de 15 % que l'entreprise collecte sur la vente, telle que **11.25** ou **73.20**. Le vendeur reçoit 85 % de la valeur PRICEPAID.
SALETIME	TIMESTAMP	Date complète et heure auxquelles la vente a été réalisée, telles que **2008-05-24 06:21:47** .

Bonnes pratiques Amazon Redshift

Vous trouverez ensuite les bonnes pratiques pour planifier une preuve de concept, concevoir des tableaux, charger des données dans des tables et écrire des requêtes pour Amazon Redshift, ainsi qu'une discussion sur la manière de travailler avec Amazon Redshift Advisor.

Amazon Redshift est différent des autres systèmes de base de données SQL. Pour profiter pleinement des avantages de l'architecture Amazon Redshift, vous devez spécifiquement concevoir, construire et charger vos tables pour utiliser le traitement massivement parallèle, le stockage de données en colonnes et la compression de données en colonnes. Si vos temps de chargement des données et d'exécution des requêtes sont plus longs qu'escompté, il se peut que vous ayez ignoré certaines informations clés.

Si vous êtes un développeur de base de données SQL expérimenté, nous vous recommandons vivement de consulter cette rubrique avant de commencer à développer votre entrepôt de données Amazon Redshift.

Si vous êtes novice dans le développement de bases de données SQL, cette rubrique n'est pas le meilleur endroit pour commencer. Nous vous recommandons de commencer par lire les commandes Exécuter pour définir et utiliser une base de données dans votre entrepôt de données dans le guide de démarrage Amazon Redshift, et d'essayer les exemples vous-même.

Cette rubrique fournit une vue d'ensemble des principes de développement les plus importants, ainsi que des conseils spécifiques, exemples et bonnes pratiques pour mettre en œuvre ces principes. Aucune pratique unique ne peut s'appliquer à toutes les applications. Évaluez toutes vos options avant de finaliser une conception de base de données. Pour plus d'informations, consultez Optimisation automatique des tables, Chargement de données dans Amazon Redshift, Réglage des performances des requêtes et les chapitres de référence.

Rubriques

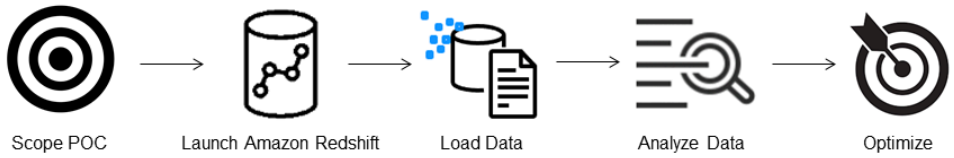
- Réaliser une preuve de concept (POC) pour Amazon Redshift
- Bonnes pratiques Amazon Redshift pour la conception de tables
- Bonnes pratiques de chargement des données sur Amazon Redshift
- Bonnes pratiques Amazon Redshift pour la conception de requêtes
- Suivez les recommandations d'Amazon Redshift Advisor

Réaliser une preuve de concept (POC) pour Amazon Redshift

Amazon Redshift est un entrepôt de données cloud populaire, qui propose un service cloud entièrement géré qui s'intègre au lac de données Amazon Simple Storage Service, aux flux en temps réel, aux flux de travail d'apprentissage automatique (ML), aux flux de travail transactionnels d'une entreprise, etc. Les sections suivantes vous guident tout au long du processus de validation de principe (POC) sur Amazon Redshift. Les informations présentées ici vous aident à définir des objectifs pour votre POC et à tirer parti des outils qui peuvent automatiser le provisionnement et la configuration des services pour votre POC.

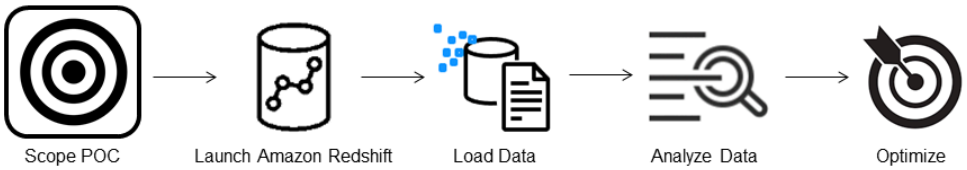
> **Note**
>
> Pour obtenir une copie de ces informations au format PDF, cliquez sur le lien Run your own Redshift POC sur la page des ressources Amazon Redshift.

Lorsque vous effectuez un POC d'Amazon Redshift, vous testez, prouvez et adoptez des fonctionnalités telles que des fonctionnalités best-in-class de sécurité, une mise à l'échelle élastique, une intégration et une ingestion faciles, ainsi que des options d'architecture de données décentralisées flexibles.



Suivez ces étapes pour réussir un POC.

Étape 1 : Déterminez le périmètre de votre POC



Lorsque vous effectuez un POC, vous pouvez choisir d'utiliser vos propres données ou d'utiliser des ensembles de données d'analyse comparative. Lorsque vous choisissez vos propres données, vous exécutez vos propres requêtes sur ces données. Avec les données d'analyse comparative, des exemples de requêtes sont fournis avec le point de référence. Consultez Utiliser des exemples de jeux de données pour plus de détails si vous n'êtes pas encore prêt à effectuer un POC avec vos propres données.

En général, nous recommandons d'utiliser deux semaines de données pour un POC Amazon Redshift.

Commencez par effectuer les opérations suivantes :

1. Identifiez vos exigences opérationnelles et fonctionnelles, puis revenez en arrière. Les exemples les plus courants sont les suivants : amélioration des performances, réduction des coûts, test d'une nouvelle charge de travail ou d'une nouvelle fonctionnalité, ou comparaison entre Amazon Redshift et un autre entrepôt de données.
2. Fixez des objectifs spécifiques qui deviennent les critères de réussite du POC. Par exemple, pour des performances plus rapides, dressez une liste des cinq principaux processus que vous souhaitez accélérer et incluez les durées d'exécution actuelles ainsi que la durée d'exécution requise. Il peut s'agir de rapports, de requêtes, de processus ETL, d'ingestion de données ou de tout autre problème actuel.
3. Identifiez la portée et les artefacts spécifiques nécessaires pour exécuter les tests. Quels ensembles de données devez-vous migrer ou ingérer en permanence dans Amazon Redshift, et quels sont les requêtes et les processus nécessaires pour exécuter les tests en fonction des critères de réussite ? Il existe deux façons de procéder :

   Apportez vos propres données

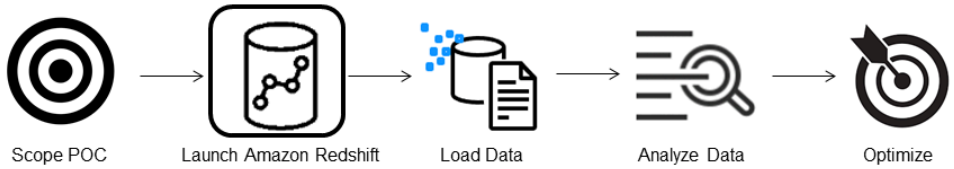
   - Pour tester vos propres données, établissez la liste minimale viable d'artefacts de données nécessaires pour tester vos critères de réussite. Par exemple, si votre entrepôt de données actuel compte 200 tables, mais que les rapports que vous souhaitez tester n'en ont besoin que de 20, votre POC peut être exécuté plus rapidement en utilisant uniquement le plus petit sous-ensemble de tables.

   Utiliser des exemples de jeux de données

   - Si vos propres ensembles de données ne sont pas prêts, vous pouvez toujours commencer à effectuer un POC sur Amazon Redshift en utilisant les ensembles de données de référence tels

que TPC-DS ou TPC-H et en exécutant des exemples de requêtes d'analyse comparative pour exploiter la puissance d'Amazon Redshift. Ces ensembles de données sont accessibles depuis votre entrepôt de données Amazon Redshift une fois celui-ci créé. Pour obtenir des instructions détaillées sur la façon d'accéder à ces ensembles de données et des exemples de requêtes, consultezÉtape 2 : Lancez Amazon Redshift.

Étape 2 : Lancez Amazon Redshift



Amazon Redshift vous permet d'obtenir plus rapidement des informations grâce à un entrepôt de données cloud rapide, simple et sécurisé à grande échelle. Vous pouvez démarrer rapidement en lançant votre entrepôt sur la console Redshift Serverless et passer des données aux informations en quelques secondes. Avec Redshift Serverless, vous pouvez vous concentrer sur les résultats de votre entreprise sans vous soucier de la gestion de votre entrepôt de données.

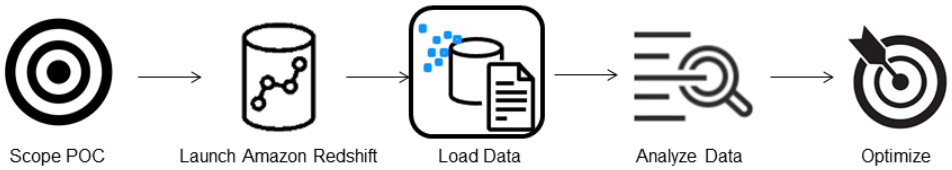
Configurer Amazon Redshift sans serveur

La première fois que vous utilisez Redshift Serverless, la console vous guide à travers les étapes nécessaires au lancement de votre entrepôt. Vous pourriez également être éligible à un crédit correspondant à votre utilisation de Redshift Serverless sur votre compte. Pour plus d'informations sur le choix d'un essai gratuit, consultez la rubrique Essai gratuit d'Amazon Redshift. Suivez les étapes décrites dans la section Création d'un entrepôt de données avec Redshift Serverless du guide de démarrage Amazon Redshift pour créer un entrepôt de données avec Redshift Serverless. Si vous n'avez pas de jeu de données à charger, le guide explique également comment charger un exemple de jeu de données.

Si vous avez déjà lancé Redshift Serverless dans votre compte, suivez les étapes décrites dans la section Création d'un groupe de travail avec un espace de noms dans le guide de gestion Amazon Redshift. Une fois votre entrepôt disponible, vous pouvez choisir de charger les exemples de données disponibles dans Amazon Redshift. Pour plus d'informations sur l'utilisation de l'éditeur de requêtes Amazon Redshift v2 pour charger des données, consultez la section Chargement d'exemples de données dans le guide de gestion Amazon Redshift.

Si vous apportez vos propres données au lieu de charger l'exemple d'ensemble de données, consultezÉtape 3 : Chargez vos données.

Étape 3 : Chargez vos données



Après avoir lancé Redshift Serverless, l'étape suivante consiste à charger vos données pour le POC. Que vous téléchargiez un simple fichier CSV, que vous ingériez des données semi-structurées depuis S3 ou que vous diffusiez directement des données, Amazon Redshift offre la flexibilité nécessaire pour déplacer rapidement et facilement les données depuis la source vers les tables Amazon Redshift.

Choisissez l'une des méthodes suivantes pour charger vos données.

Téléchargez un fichier local

Pour une ingestion et une analyse rapides, vous pouvez utiliser l'éditeur de requêtes Amazon Redshift v2 pour charger facilement des fichiers de données depuis votre bureau local. Il a la capacité de traiter des fichiers dans différents formats tels que CSV, JSON, AVRO, PARQUET, ORC, etc. Pour permettre à vos utilisateurs, en tant qu'administrateur, de charger des données depuis un poste de travail local à l'aide de l'éditeur de requêtes v2, vous devez spécifier un compartiment Amazon S3 commun, et le compte utilisateur doit être configuré avec les autorisations appropriées. Vous pouvez suivre le chargement des données en toute simplicité et en toute sécurité dans Amazon Redshift à l'aide de Query Editor V2 pour step-by-step obtenir des conseils.

Charger un fichier Amazon S3

Pour charger des données depuis un compartiment Amazon S3 dans Amazon Redshift, commencez par utiliser la commande COPY, en spécifiant l'emplacement Amazon S3 source et la table Amazon Redshift cible. Assurez-vous que les rôles et autorisations IAM sont correctement configurés pour autoriser Amazon Redshift à accéder au compartiment Amazon S3 désigné. Suivez le didacticiel : Chargement de données depuis Amazon S3 pour step-by-step obtenir des conseils. Vous pouvez également choisir l'option Charger les données dans l'éditeur de requêtes v2 pour charger directement les données depuis votre compartiment S3.

Ingestion continue des données

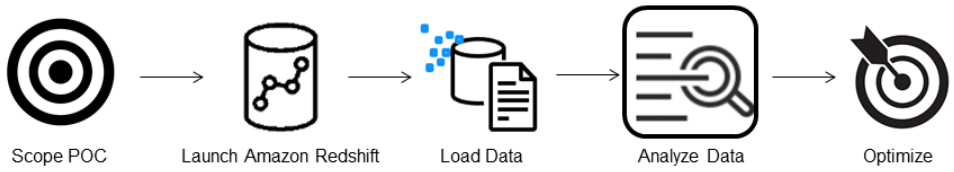
Autocopy (en version préliminaire) est une extension de la commande COPY qui automatise le chargement continu des données à partir des compartiments Amazon S3. Lorsque vous créez une tâche de copie, Amazon Redshift détecte la création de nouveaux fichiers Amazon S3 dans un chemin spécifié, puis les charge automatiquement sans votre intervention. Amazon Redshift assure le suivi des fichiers chargés afin de vérifier qu'ils ne sont chargés qu'une seule fois. Pour obtenir des instructions sur la création de tâches de copie, voir TÂCHE DE COPIE

> **Note**
>
> La copie automatique est actuellement en version préliminaire et n'est prise en charge que dans des clusters provisionnés en particulier. Régions AWS Pour créer un cluster de prévisualisation à des fins d'autocopie, voirCréez une intégration d'événements S3 pour copier automatiquement des fichiers depuis des compartiments Amazon S3.

Chargez vos données de streaming

L'ingestion du streaming permet une ingestion rapide et à faible latence des données de flux provenant d'Amazon Kinesis Data Streams et d'Amazon Managed Streaming for Apache Kafka vers Amazon Redshift. L'ingestion du streaming par Amazon Redshift utilise une vue matérialisée, qui est mise à jour directement à partir du flux à l'aide de l'actualisation automatique. La vue matérialisée est mappée à la source de données du flux. Vous pouvez effectuer des filtrages et des agrégations sur les données du flux dans le cadre de la définition de la vue matérialisée. Pour obtenir step-by-step des conseils sur le chargement de données depuis un flux, consultez Getting started with Amazon Kinesis Data Streams ou Getting started with Amazon Managed Streaming for Apache Kafka.

Étape 4 : Analysez vos données



Après avoir créé votre groupe de travail et votre espace de noms Redshift Serverless, et après avoir chargé vos données, vous pouvez immédiatement exécuter des requêtes en ouvrant l'éditeur de

requêtes v2 depuis le panneau de navigation de la console Redshift Serverless. Vous pouvez utiliser l'éditeur de requêtes v2 pour tester les fonctionnalités ou les performances des requêtes par rapport à vos propres ensembles de données.

Requête à l'aide de l'éditeur de requêtes Amazon Redshift v2

Vous pouvez accéder à l'éditeur de requêtes v2 depuis la console Amazon Redshift. Consultez Simplifiez votre analyse de données avec l'éditeur de requêtes Amazon Redshift v2 pour un guide complet sur la façon de configurer, connecter et exécuter des requêtes avec l'éditeur de requêtes v2.

Sinon, si vous souhaitez exécuter un test de charge dans le cadre de votre POC, vous pouvez le faire en suivant les étapes suivantes pour installer et exécuter Apache JMeter.

Exécuter un test de charge à l'aide d'Apache JMeter

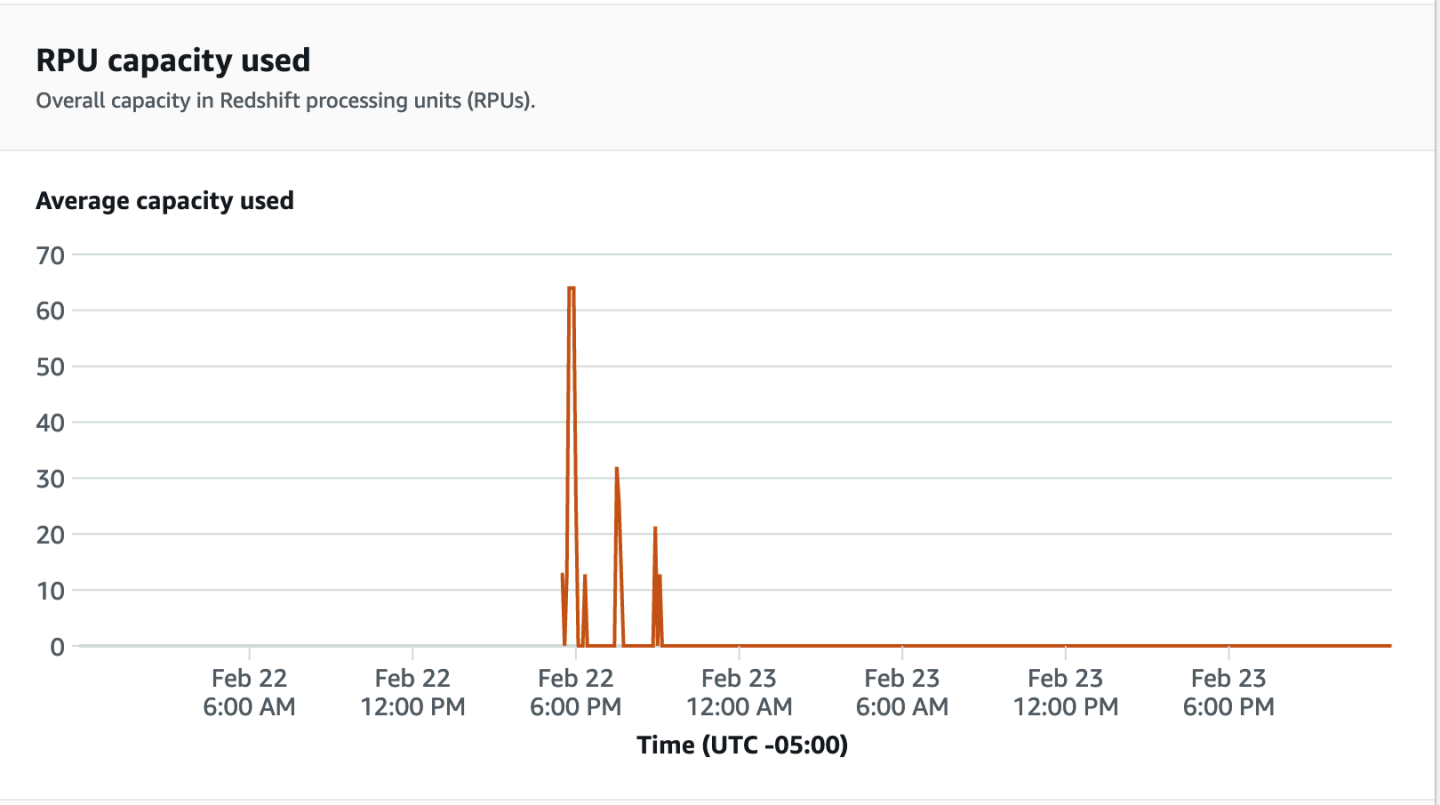
Pour effectuer un test de charge afin de simuler « N » utilisateurs soumettant des requêtes simultanément à Amazon Redshift, vous pouvez utiliser JMeter Apache, un outil open source basé sur Java.

Pour installer et configurer Apache JMeter afin qu'il s'exécute sur votre groupe de travail Redshift Serverless, suivez les instructions de la section Automatiser les tests de charge Amazon Redshift avec l'Analytics Automation Toolkit. AWS Il utilise le kit d'outils AWS Analytics Automation (AAA), un utilitaire open source permettant de déployer dynamiquement des solutions Redshift, pour lancer automatiquement ces ressources. Si vous avez chargé vos propres données dans Amazon Redshift, veillez à exécuter l'option Étape #5 — Personnaliser le SQL, afin de vous assurer de fournir les instructions SQL appropriées que vous souhaitez tester par rapport à vos tables. Testez chacune de ces instructions SQL une fois à l'aide de l'éditeur de requêtes v2 pour vous assurer qu'elles s'exécutent sans erreur.

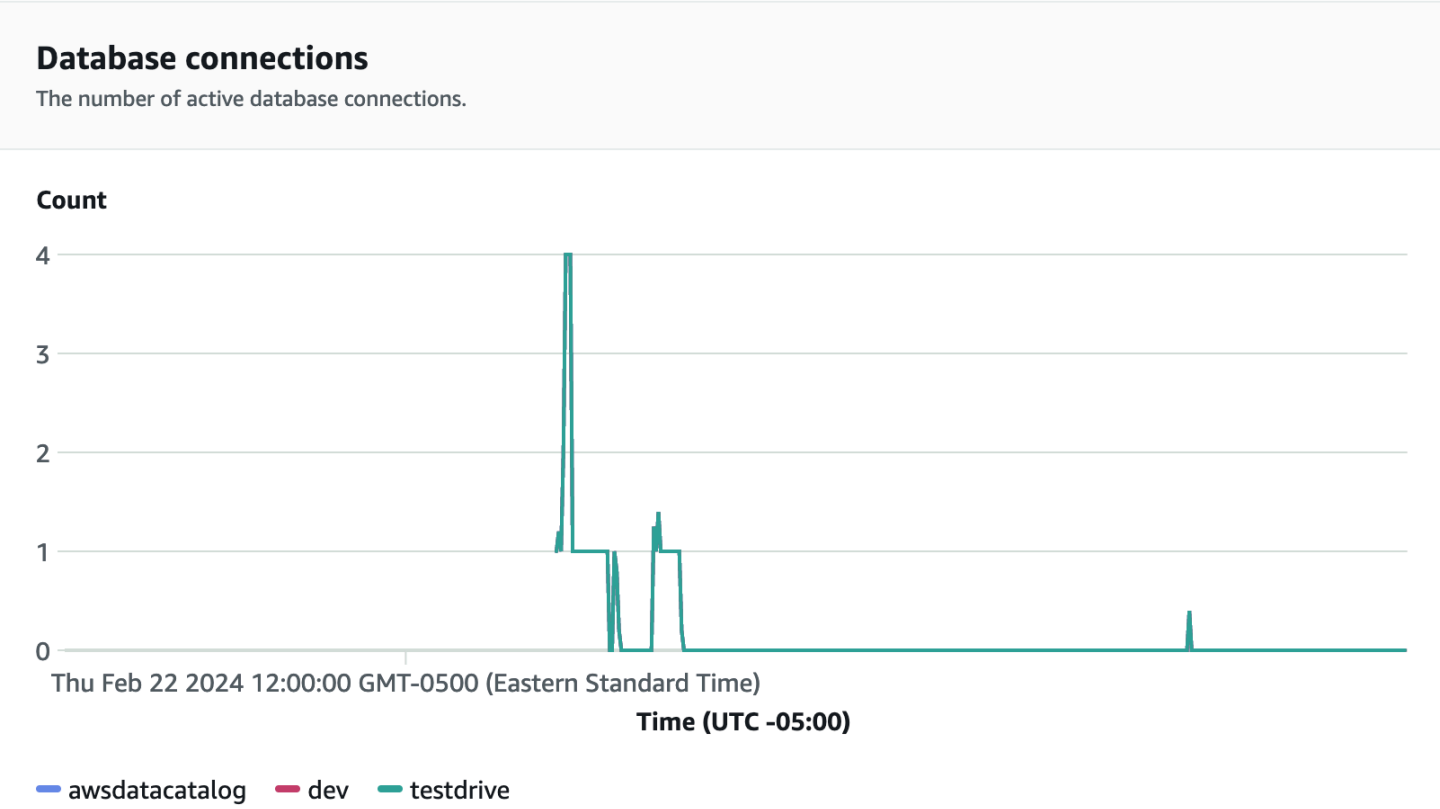
Une fois que vous avez terminé de personnaliser vos instructions SQL et de finaliser votre plan de test, enregistrez et exécutez votre plan de test par rapport à votre groupe de travail Redshift Serverless. Pour suivre la progression de votre test, ouvrez la console Redshift Serverless, accédez à Surveillance des requêtes et des bases de données, choisissez l'onglet Historique des requêtes et consultez les informations relatives à vos requêtes.

Pour les mesures de performance, choisissez l'onglet Performances de la base de données sur la console Redshift Serverless, pour surveiller les mesures telles que les connexions aux bases de données et l'utilisation du processeur. Vous pouvez consulter ici un graphique pour surveiller la capacité RPU utilisée et observer comment Redshift Serverless évolue automatiquement pour

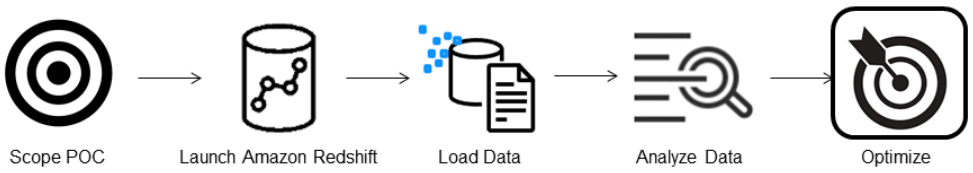
répondre aux demandes de charge de travail simultanées pendant que le test de charge est en cours d'exécution sur votre groupe de travail.



Les connexions aux bases de données constituent un autre indicateur utile à surveiller lors de l'exécution du test de charge pour voir comment votre groupe de travail gère de nombreuses connexions simultanées à un moment donné afin de répondre aux demandes croissantes de charge de travail.



Étape 5 : Optimisation



Amazon Redshift permet à des dizaines de milliers d'utilisateurs de traiter des exaoctets de données chaque jour et d'optimiser leurs charges de travail d'analyse en proposant une variété de configurations et de fonctionnalités adaptées à des cas d'utilisation individuels. Lorsqu'ils choisissent entre ces options, les clients recherchent des outils qui les aident à déterminer la configuration d'entrepôt de données la plus optimale pour prendre en charge leur charge de travail Amazon Redshift.

Essai routier

Vous pouvez utiliser Test Drive pour rejouer automatiquement votre charge de travail existante sur des configurations potentielles et analyser les résultats correspondants afin d'évaluer la cible

optimale vers laquelle migrer votre charge de travail. Consultez Trouver la meilleure configuration Amazon Redshift pour votre charge de travail à l'aide de Redshift Test Drive pour plus d'informations sur l'utilisation de Test Drive pour évaluer différentes configurations Amazon Redshift.

Bonnes pratiques Amazon Redshift pour la conception de tables

Lorsque vous planifiez votre base de données, certaines décisions clés en matière de conception de table influencent fortement les performances globales des requêtes. Ces choix de conception ont également un effet significatif sur les besoins en stockage, lesquels, à leur tour, affectent les performances des requêtes en réduisant le nombre d'opérations d'I/O et la mémoire requise pour traiter les requêtes.

Cette section résume les plus importantes décisions de conception et les bonnes pratiques d'optimisation des performances des requêtes. Optimisation automatique des tables fournit des explications et des exemples plus détaillés d'options de conception de table.

Rubriques

- Choix de la meilleure clé de tri
- Choisir le meilleur style de distribution
- Laissez COPY choisir les encodages de compression
- Définir les contraintes de clé primaire et de clé étrangère
- Utiliser la plus petite taille de colonne possible
- Utiliser les types de données date/time pour les colonnes de date

Choix de la meilleure clé de tri

Amazon Redshift stocke vos données sur le disque dans un ordre trié selon la clé de tri. L'optimiseur de requête Amazon Redshift utilise l'ordre de tri lorsqu'il détermine les plans de requête optimaux.

> **Note**
>
> Lorsque vous utilisez l'optimisation automatique des tables, vous n'avez pas besoin de choisir la clé de tri de votre table. Pour plus d'informations, consultez Optimisation automatique des tables.

Voici quelques suggestions pour la meilleure approche :

- Pour qu'Amazon Redshift choisisse l'ordre de tri approprié, spécifiez AUTO pour la clé de tri.
- Si les données récentes sont interrogées le plus fréquemment, spécifiez la colonne d'horodatage en tant que colonne principale de la clé de tri.

  Les requêtes sont plus efficaces, car elles peuvent ignorer des blocs entiers qui ne relèvent pas de la plage de temps.
- Si vous effectuez le filtrage par plage ou par égalité sur une seule colonne, spécifiez cette colonne comme clé de tri.

  Amazon Redshift peut omettre la lecture de blocs entiers de données pour cette colonne. Il peut le faire car il assure le suivi des valeurs de colonne minimales et maximales stockées sur chaque bloc et peut ignorer les blocs qui ne s'appliquent pas à la plage de prédicats.
- Si vous effectuez fréquemment la jointure d'une table, spécifiez la colonne de jointure à la fois comme clé de tri et comme clé de distribution.

  Cela permet à l'optimiseur de requête de choisir une sorte de jointure de fusion triée au lieu d'une jointure de hachage plus lente. Étant donné que les données sont déjà triées sur la clé de jointure, l'optimiseur de requête peut contourner la phase de tri de la jointure de fusion triée.

Choisir le meilleur style de distribution

Lorsque vous exécutez une requête, l'optimiseur de requête redistribue les lignes sur les nœuds de calcul en fonction des besoins afin d'effectuer les jointures et les agrégations. Le choix d'un style de distribution de table a pour objectif de minimiser l'impact de l'étape de redistribution en plaçant les données où elles doivent être avant que la requête ne soit exécutée.

> **Note**
>
> Lorsque vous utilisez l'optimisation automatique des tables, vous n'avez pas besoin de choisir le mode de distribution de votre table. Pour plus d'informations, consultez Optimisation automatique des tables.

Voici quelques suggestions pour la meilleure approche :

1. Distribuez la table des faits et une table de dimension sur leurs colonnes communes.

Votre table de faits peut n'avoir qu'une seule clé de distribution. Toutes les tables qui sont jointes à une autre clé de distribution ne sont pas colocalisées avec la table des faits. Choisissez une dimension pour colocaliser en fonction de la fréquence à laquelle elle est jointe et de la taille des lignes de jointure. Désignez la clé primaire de la table de dimension et la clé étrangère correspondante de la table des faits comme DISTKEY.

2. Choisissez la plus grande dimension en fonction de la taille de l'ensemble de données filtré.

   Comme seules les lignes utilisées dans la jointure doivent être distribuées, prenez en compte la taille du jeu de données après filtrage, et non la taille de la table.

3. Sélectionnez une colonne avec cardinalité élevée dans l'ensemble de résultats filtré.

   Si vous distribuez une table des ventes sur une colonne date, par exemple, vous obtiendrez probablement une distribution des données assez uniforme, à moins que la plupart de vos ventes ne soient saisonnières. Cependant, si vous utilisez généralement un prédicat à plage restreinte pour filtrer sur une période de dates étroite, la plupart des lignes filtrées se trouvent sur un ensemble limité de tranches et la charge de travail des requêtes est biaisée.

4. Modifiez certaines tables de dimension pour utiliser la distribution ALL.

   Si une table de dimension ne peut pas être colocalisée avec la table des faits ou d'autres tables de jointure importantes, il vous est possible d'améliorer considérablement les performances des requêtes en distribuant la totalité de la table sur tous les nœuds. La distribution ALL multiplie les besoins en espace de stockage et augmente les temps de chargement et les opérations de maintenance. Vous devez donc évaluer tous les facteurs avant de choisir la distribution ALL.

Pour demander à Amazon Redshift de choisir le style de distribution approprié, spécifiez `AUTO` pour le style de distribution.

Pour plus d'informations sur le choix des styles de distribution, consultez Distribution des données pour l'optimisation des requêtes.

Laissez COPY choisir les encodages de compression

Vous pouvez spécifier les encodages de compression lorsque vous créez une table, mais, dans la plupart des cas, la compression automatique donne les meilleurs résultats.

ENCODE AUTO est la valeur par défaut pour les tables. Lorsqu'une table est définie sur ENCODE AUTO, Amazon Redshift gère automatiquement l'encodage de compression pour toutes les colonnes de la table. Pour plus d'informations, consultez CREATE TABLE et ALTER TABLE.

La commande COPY analyse vos données et applique automatiquement les encodages de compression sur une table vide dans le cadre de l'opération de chargement.

La compression automatique équilibre les performances globales lors du choix des encodages de compression. Les analyses à plage restreinte peuvent mal s'exécuter si les colonnes de clé de tri sont beaucoup plus compressées que les autres colonnes de la même requête. Par conséquent, la compression automatique choisit un encodage de compression moins efficace pour que les colonnes de clé de tri demeurent équilibrées avec les autres colonnes.

Supposons que la clé de tri de la table soit de type date ou timestamp, et que la table utilise de nombreuses colonnes de type varchar. Dans ce cas, vous pourrez obtenir de meilleures performances en ne compressant pas du tout la colonne de la clé de tri. Exécutez la commande ANALYZE COMPRESSION sur la table, puis utilisez les encodages pour créer une table, mais excluez l'encodage de compression pour la clé de tri.

L'encodage de compression automatique présente un coût en termes de performances, mais seulement si la table est vide et qu'elle ne possède pas déjà un encodage de compression. Pour les tables de courte durée et les tables que vous créez fréquemment, telles que les tables intermédiaires, chargez la table une fois avec la compression automatique ou exécutez la commande ANALYZE COMPRESSION. Utilisez ensuite ces encodages pour créer des tables. Vous pouvez ajouter les encodages à l'instruction CREATE TABLE, ou utiliser CREATE TABLE LIKE pour créer une table avec le même encodage.

Pour plus d'informations, consultez Chargement des tables avec compression automatique.

Définir les contraintes de clé primaire et de clé étrangère

Définissez les contraintes de clé primaire et de clé étrangère entre les tables chaque fois que nécessaire. Même si elles n'ont qu'une valeur informative, l'optimiseur de requête utilise ces contraintes pour générer des plans de requête plus efficaces.

Ne définissez pas de contraintes de clé primaire et de clé étrangère à moins que votre application ne les applique. Amazon Redshift n'applique pas les contraintes d'unicité, de clé primaire et de clé étrangère.

Consultez Contraintes liées aux tables pour obtenir des informations supplémentaires sur la manière dont Amazon Redshift utilise les contraintes.

Utiliser la plus petite taille de colonne possible

Ne vous habituez pas à utiliser la taille de colonne maximale pour des raisons de commodité.

Au lieu de cela, prenez en compte les valeurs les plus grandes que vous êtes susceptible de stocker dans vos colonnes et dimensionnez ces dernières en conséquence. Par exemple, une colonne CHAR pour stocker les abréviations des États et territoires américains utilisées par le bureau de poste doit uniquement être CHAR (2).

Utiliser les types de données date/time pour les colonnes de date

Amazon Redshift stocke les données DATE et TIMESTAMP de manière plus efficace que CHAR ou VARCHAR, ce qui permet d'améliorer les performances des requêtes. Utilisez le type de données DATE ou TIMESTAMP, en fonction de la résolution dont vous avez besoin, plutôt qu'un type caractère lors du stockage d'informations de type date/time. Pour plus d'informations, consultez Types datetime.

Bonnes pratiques de chargement des données sur Amazon Redshift

Le chargement de très grands ensembles de données peut prendre du temps et consommer beaucoup de ressources de calcul. La façon dont vos données sont chargées peut aussi affecter les performances des requêtes. Cette section présente les bonnes pratiques pour charger efficacement les données à l'aide des commandes COPY, des insertions en bloc et des tables intermédiaires.

Rubriques

- Découvrez comment charger des données grâce à un didacticiel
- Utiliser une commande COPY pour charger les données
- Utiliser une seule commande COPY pour charger à partir de plusieurs fichiers
- Chargement de fichiers de données
- Compression de vos fichiers de données
- Vérifier les fichiers de données avant et après un chargement
- Utiliser une insertion multiligne

- Utiliser une insertion en bloc
- Charger les données dans l'ordre de la clé de tri
- Charger les données en blocs séquentiels
- Utiliser les tables chronologiques
- Planifier la maintenance Windows

Découvrez comment charger des données grâce à un didacticiel

Didacticiel : chargement des données à partir d'Amazon S3 vous guide du début à la fin à travers les étapes permettant de télécharger des données vers un compartiment Amazon S3, puis d'utiliser la commande COPY pour charger les données dans vos tables. Le didacticiel inclut une aide à la résolution des erreurs de chargement et compare les différences de performance entre le chargement à partir d'un seul fichier et le chargement à partir de plusieurs fichiers.

Utiliser une commande COPY pour charger les données

La commande COPY charge des données en parallèle depuis Amazon S3, Amazon EMR, Amazon DynamoDB ou plusieurs sources de données sur des hôtes distants. COPY charge de grandes quantités de données beaucoup plus efficacement que les instructions INSERT, et stocke aussi les données plus efficacement.

Pour plus d'informations sur l'utilisation de la commande COPY, consultez Chargement des données à partir d'Amazon S3 et Chargement de données à partir d'une table Amazon DynamoDB.

Utiliser une seule commande COPY pour charger à partir de plusieurs fichiers

Amazon Redshift charge automatiquement en parallèle à partir de plusieurs fichiers de données compressés. Vous pouvez spécifier les fichiers à charger en utilisant un préfixe d'objet Amazon S3 ou un fichier manifeste.

Toutefois, si vous utilisez plusieurs commandes COPY simultanées pour charger une table à partir de plusieurs fichiers, Amazon Redshift est obligé d'effectuer un chargement sérialisé. Ce type de chargement est beaucoup plus lent et requiert un processus VACUUM à la fin si la table possède une colonne de tri définie. Pour plus d'informations sur l'utilisation de COPY pour charger les données en parallèle, consultez Chargement des données à partir d'Amazon S3.

Chargement de fichiers de données

Les fichiers de données sources se présentent sous différents formats et utilisent divers algorithmes de compression. Lorsque vous chargez des données à l'aide de la commande COPY, Amazon Redshift charge tous les fichiers référencés par le préfixe de compartiment Amazon S3. (Le préfixe est une chaîne de caractères située au début du nom de clé d'objet.) Si le préfixe fait référence à plusieurs fichiers ou à des fichiers fractionnables, Amazon Redshift charge les données en parallèle, tirant parti de l'architecture MPP d'Amazon Redshift. La charge de travail est alors répartie entre les nœuds du cluster. En revanche, lorsque vous chargez des données à partir d'un fichier non fractionnable, Amazon Redshift est forcé d'effectuer un chargement sérialisé, ce qui est beaucoup plus long. Les sections suivantes décrivent la méthode recommandée pour charger différents types de fichiers dans Amazon Redshift, en fonction de leur format et de leur compression.

Chargement de données à partir de fichiers fractionnables

Les fichiers suivants peuvent être automatiquement fractionnés pendant le chargement de leurs données :

- Fichier CSV non compressé
- Fichier en colonnes (Parquet/ORC)

Amazon Redshift fractionne automatiquement les fichiers de 128 Mo ou plus en fragments. Les fichiers en colonnes, en particulier Parquet et ORC, ne sont pas fractionnés s'ils font moins de 128 Mo. Pour charger les données, Redshift utilise des tranches fonctionnant en parallele. Cela autorise des chargements rapides.

Chargement de données à partir de fichiers non fractionnables

Certains types de fichiers, comme JSON ou CSV, ne sont pas automatiquement fractionnés lorsqu'ils sont compressés avec d'autres algorithmes de compression, tels que GZIP. Dans ce cas, nous vous recommandons de fractionner manuellement les données en plusieurs petits fichiers dont les tailles sont aussi proches que possible les unes des autres, de 1 Mo à 1 Go après compression. De plus, faites en sorte que le nombre de fichiers soit un multiple du nombre de tranches présentes dans votre cluster. Pour en savoir plus sur la façon de fractionner vos données en plusieurs fichiers et pour voir des exemples de chargement de données à l'aide de COPY, consultez Chargement des données à partir d'Amazon S3.

Compression de vos fichiers de données

Lorsque vous souhaitez compresser des fichiers de chargement volumineux, nous vous recommandons d'utiliser gzip, lzop, bzip2 ou Zstandard pour les compresser et diviser les données en plusieurs fichiers plus petits.

Spécifiez l'option GZIP, LZOP ou ZSTD à l'aide de la commande COPY. BZIP2 Cet exemple charge la table TIME à partir d'un fichier lzop délimité par une barre verticale.

```
copy time
from 's3://amzn-s3-demo-bucket/data/timerows.lzo'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
lzop
delimiter '|';
```

Il existe des cas où vous n'avez pas à diviser des fichiers de données non compressés. Pour plus d'informations sur le fractionnement de vos données et obtenir des exemples sur l'utilisation de COPY pour charger les données, consultez Chargement des données à partir d'Amazon S3.

Vérifier les fichiers de données avant et après un chargement

Avant de charger des données à partir d'Amazon S3, commencez par vérifier que votre compartiment Amazon S3 contient tous les bons fichiers, et uniquement ces fichiers. Pour de plus amples informations, veuillez consulter Vérification de la présence des fichiers corrects dans votre compartiment.

Une fois l'opération de chargement terminée, interrogez la table système STL_LOAD_COMMITS pour vérifier que les fichiers attendus ont été chargés. Pour plus d'informations, consultez Vérification que les données ont été chargées correctement.

Utiliser une insertion multiligne

Si une commande COPY n'est pas une option et que vous avez besoin d'insertions SQL, utilisez une insertion multiligne chaque fois que possible. La compression des données est inefficace lorsque vous ajoutez uniquement une seule ligne ou quelques lignes à la fois.

Les insertions multiligne améliorent les performances en traitant par lots à une série d'insertions. L'exemple suivant insère trois lignes dans une table de quatre colonnes à l'aide d'une seule instruction INSERT. Il s'agit toujours d'une petite insertion, montrée simplement pour illustrer la syntaxe d'une insertion multiligne.

```
insert into category_stage values
(default, default, default, default),
(20, default, 'Country', default),
(21, 'Concerts', 'Rock', default);
```

Pour obtenir des détails et des exemples, consultez INSERT.

Utiliser une insertion en bloc

Utilisez une opération d'insertion en bloc avec une clause SELECT pour une insertion de données hautes performances.

Utilisez les commandes INSERT et CREATE TABLE AS lorsque vous avez besoin de déplacer les données ou un sous-ensemble de données d'une table vers une autre.

Par exemple, l'instruction INSERT suivante sélectionne toutes les lignes de la table CATEGORY et les insère dans la table CATEGORY_STAGE.

```
insert into category_stage
(select * from category);
```

L'exemple suivant crée CATEGORY_STAGE comme copie de CATEGORY et insère toutes les lignes de CATEGORY dans CATEGORY_STAGE.

```
create table category_stage as
select * from category;
```

Charger les données dans l'ordre de la clé de tri

Chargez les données dans l'ordre de la clé de tri pour éviter la nécessité d'une opération VACUUM.

Si chaque lot de nouvelles données suit les lignes existantes de votre table, vos données sont correctement stockées selon l'ordre de tri et vous n'avez pas à exécuter une opération VACUUM. Vous n'avez pas besoin de trier au préalable les lignes de chaque chargement, car COPY trie chaque lot de données entrantes au chargement.

Par exemple, supposons que vous chargiez les données chaque jour en fonction de l'activité quotidienne. Si votre clé de tri est une colonne de type timestamp, vos données sont stockées dans l'ordre de tri. Cet ordre est utilisé, car les données du jour en cours sont toujours ajoutées à la fin des données de la journée précédente. Pour de plus amples informations, veuillez consulter Chargez vos

données dans l'ordre des clés de tri. Pour plus d'informations sur les opérations de purge, consultez Purge des tables.

Charger les données en blocs séquentiels

Si vous devez ajouter une grande quantité de données, chargez les données en blocs séquentiels selon l'ordre de tri afin d'éliminer la nécessité d'une opération VACUUM.

Par exemple, supposons que vous deviez charger une table avec les événements compris entre janvier 2017 et décembre 2017. En supposant que chaque mois est dans un seul fichier, chargez les lignes pour janvier, puis février, et ainsi de suite. Votre table est totalement triée lorsque votre chargement est terminé et vous n'avez pas besoin d'exécuter une opération VACUUM. Pour plus d'informations, consultez Utiliser les tables chronologiques.

Lors du chargement de très grands ensembles de données, l'espace nécessaire pour le tri peut dépasser l'espace disponible total. En chargeant les données par blocs plus petits, vous utilisez beaucoup moins d'espace de tri intermédiaire pendant chaque chargement. En outre, le chargement de blocs plus petits facilite le redémarrage si la commande COPY échoue et est annulée.

Utiliser les tables chronologiques

Si vos données ont une période de conservation fixe, vous pouvez les organiser sous forme de séquence de tables chronologiques. Dans une séquence de ce type, chaque table est identique mais contient des données provenant de différentes plages horaires.

Vous pouvez facilement supprimer les données anciennes en exécutant une commande DROP TABLE sur les tables correspondantes. Cette approche est beaucoup plus rapide que l'exécution d'une opération DELETE à grande échelle et vous évite aussi de devoir exécuter ensuite une opération VACUUM pour récupérer de l'espace. Pour masquer le fait que les données sont stockées dans des tables différentes, vous pouvez créer une vue UNION ALL. Lorsque vous supprimez des données anciennes, affinez votre vue UNION ALL pour retirer les tables supprimées. De même, lorsque vous chargez de nouvelles périodes dans les nouvelles tables, ajoutez ces nouvelles tables à la vue. Pour signaler à l'optimiseur de ne pas effectuer l'analyse sur les tables qui ne correspondent pas au filtre de requête, votre définition de vue filtre la plage de dates qui correspond à chaque table.

Évitez d'avoir trop de tables dans la vue UNION ALL. Chaque table supplémentaire ajouter un petit temps de traitement à la requête. Les tables n'ont pas besoin d'utiliser la même période de temps. Vous pouvez avoir des tables pour des périodes de temps différentes, par exemple quotidiennes, mensuelles et annuelles.

Si vous utilisez des tables chronologiques avec une colonne de type timestamp pour la clé de tri, vous chargez efficacement vos données dans l'ordre de la clé de tri. Cela vous évite d'avoir à exécuter une opération VACUUM pour trier à nouveau les données. Pour de plus amples informations, veuillez consulter Chargez vos données dans l'ordre des clés de tri.

Planifier la maintenance Windows

Si une maintenance planifiée se produit pendant qu'une requête est en cours d'exécution, la requête est arrêtée et annulée, et vous devez la redémarrer. Planifiez les opérations de longue durée, telles que les chargements de données volumineux ou l'opération VACUUM, afin d'éviter les fenêtres de maintenance. Vous pouvez également réduire le risque et faciliter les redémarrages lorsqu'ils sont nécessaires, en effectuant des chargements de données sous forme d'incréments plus petits et en gérant la taille de vos opérations VACUUM. Pour plus d'informations, consultez Charger les données en blocs séquentiels et Exécution de l'opération VACUUM sur les tables.

Bonnes pratiques Amazon Redshift pour la conception de requêtes

Afin d'optimiser les performances des requêtes, suivez ces recommandations lors de la création de requêtes.

- Concevez les tables conformément aux bonnes pratiques pour fournir des bases solides aux performances des requêtes. Pour plus d'informations, consultez Bonnes pratiques Amazon Redshift pour la conception de tables.
- Évitez l'utilisation de `select *`. Incluez uniquement les colonnes dont vous avez spécifiquement besoin.
- Utilisez un Expression conditionnelle CASE pour effectuer des regroupements complexes au lieu de choisir à partir de la même table plusieurs fois.
- N'utilisez pas les jointures croisées, sauf nécessité absolue. Ces jointures sans condition de jointure se traduisent par le produit cartésien de deux tables. Les jointures croisées sont généralement exécutées comme jointures par boucle imbriquée, qui sont les types de jointures les plus lents possibles.
- Utilisez les sous-requêtes dans les cas où une table de la requête n'est utilisée que pour les conditions de prédicat et où les sous-requêtes retournent un petit nombre de lignes (inférieur à 200). L'exemple suivant utilise une sous-requête pour éviter de joindre la table LISTING.

```
select sum(sales.qtysold)
from sales
```

```
where salesid in (select listid from listing where listtime > '2008-12-26');
```

- Utilisez les prédicats afin de limiter le jeu de données autant que possible.
- Dans le prédicat, utilisez les opérateurs les moins onéreux que vous puissiez. Les opérateurs Condition de comparaison sont préférables aux opérateurs LIKE. Les opérateurs LIKE continuent d'être préférables aux opérateurs SIMILAR TO ou Opérateurs POSIX.
- Évitez l'utiliser de fonctions dans les prédicats de requête. Leur utilisation peut accroître le coût de la requête en exigeant de grands nombres de lignes pour résoudre les étapes intermédiaires de la requête.
- Si possible, utilisez la clause WHERE pour limiter l'ensemble de données. Comme le planificateur de requête peut alors utiliser l'ordre des lignes pour aider à déterminer quels enregistrements correspondent aux critères, il peut ignorer l'analyse de grands nombres de blocs de disque. Sans cela, le moteur d'exécution des requêtes doit analyser la totalité des colonnes participantes.
- Ajoutez des prédicats pour filtrer les tables qui prennent part à des jointures, même si les prédicats appliquent les mêmes filtres. La requête renvoie le même ensemble de résultats, mais Amazon Redshift est capable de filtrer les tables de jonction avant l'étape d'analyse et peut ensuite sauter efficacement l'analyse des blocs de ces tables. Les filtres redondants ne sont pas nécessaires si vous filtrez une colonne utilisée dans la condition de jonction.

  Par exemple, supposons que vous souhaitiez joindre `SALES` et `LISTING` pour trouver les ventes de billets pour les billets répertoriés après décembre, regroupés par vendeur. Les deux tables sont triées sur la date. La requête suivante joint les tables sur leur clé commune et filtre les valeurs de `listing.listtime` postérieures au 1er décembre.

  ```
  select listing.sellerid, sum(sales.qtysold)
  from sales, listing
  where sales.salesid = listing.listid
  and listing.listtime > '2008-12-01'
  group by 1 order by 1;
  ```

  Comme la clause WHERE n'inclut pas de prédicat pour `sales.saletime`, le moteur d'exécution est contraint d'analyser la totalité de la table `SALES`. Si vous savez que le filtre se traduira par le fait que moins de lignes prennent part à la jointure, ajoutez également le filtre. L'exemple suivant réduit le temps d'exécution de façon significative.

  ```
  select listing.sellerid, sum(sales.qtysold)
  from sales, listing
  where sales.salesid = listing.listid
  ```

```
and listing.listtime > '2008-12-01'
and sales.saletime > '2008-12-01'
group by 1 order by 1;
```

- Utilisez les clés de tri dans la clause GROUP BY afin que le planificateur de requête puisse utiliser l'agrégation plus efficacement. Une requête peut être éligible à une agrégation en une phase lorsque sa liste GROUP BY contient uniquement des colonnes de clé tri, l'une d'elles étant aussi la clé de distribution. Les colonnes de clé de tri de la liste GROUP BY doivent inclure la première clé de tri, puis les autres clés de tri que vous souhaitez utiliser dans l'ordre des clés de tri. Par exemple, vous pouvez parfaitement utiliser la première clé de tri, les première et deuxième clés de tri, les première, deuxième et troisième clés de tri, et ainsi de suite. Vous ne pouvez pas utiliser les première et troisième clés de tri.

  Vous pouvez confirmer l'utilisation de l'agrégation en une phase en exécutant la commande EXPLAIN et en recherchant `XN GroupAggregate` dans l'étape d'agrégation de la requête.
- Si vous utilisez les clauses GROUP BY et ORDER BY, assurez-vous d'y placer les colonnes dans le même ordre. Autrement dit, utilisez l'approche suivante :

  ```
  group by a, b, c
  order by a, b, c
  ```

  N'utilisez pas l'approche suivante :

  ```
  group by b, c, a
  order by a, b, c
  ```

Suivez les recommandations d'Amazon Redshift Advisor

Afin de vous aider à améliorer les performances de votre cluster Amazon Redshift et d'en réduire les coûts, Amazon Redshift Advisor met à votre disposition des recommandations spécifiques sur les modifications à appliquer. Advisor développe ses recommandations personnalisées en analysant les performances et les métriques d'utilisation de votre cluster. Ces recommandations personnalisées concernent les paramètres de cluster et le fonctionnement. Afin de vous aider à prioriser vos optimisations, Advisor classe les recommandations par ordre d'impact.

Advisor base ses recommandations sur les observations faites concernant les statistiques de performances ou les données de fonctionnement. Il développe des observations en exécutant des tests sur vos clusters afin de déterminer si une valeur de test se trouve dans une plage définie. Si

le résultat du test est en dehors de cette plage, Advisor génère une observation concernant votre cluster. En même temps, Advisor crée une recommandation sur la façon de ramener la valeur observée dans la plage de bonne pratique. Advisor affiche uniquement les recommandations qui doivent avoir un impact significatif sur les performances et sur le fonctionnement. Lorsque Advisor considère qu'une recommandation a été appliquée, il la supprime de votre liste de recommandations.

Par exemple, supposons que votre entrepôt de données contienne un grand nombre de colonnes de table non compressées. Dans ce cas, vous pouvez économiser sur les coûts de stockage de cluster en recréant les tables en utilisant le paramètre `ENCODE` pour spécifier la compression des colonnes. Dans un autre exemple, supposons qu'Advisor observe que votre cluster contient une quantité importante de données de table non compressées. Dans ce cas, il vous fournit le bloc de code SQL permettant de trouver les colonnes de table susceptibles d'être comprimées et les ressources qui décrivent comment compresser ces colonnes.

Régions Amazon Redshift dans lesquelles Advisor est pris en charge

La fonctionnalité Amazon Redshift Advisor n'est disponible que dans les régions suivantes : AWS

- Région USA Est (Virginie du Nord) (us-east-1)
- Région USA Est (Ohio) (us-east-2)
- Région US Ouest (Californie du Nord) (us-west-1)
- Région USA Ouest (Oregon) (us-west-2)
- Région Afrique (Le Cap) (af-south-1)
- Région Asie-Pacifique (Hong Kong) (ap-east-1)
- Région Asie-Pacifique (Hyderabad) (ap-south-2)
- Région Asie-Pacifique (Jakarta) (ap-southeast-3)
- Région Asie-Pacifique (Melbourne) (ap-southeast-4)
- Région Asie-Pacifique (Malaisie) (ap-southeast-5)
- Région Asie-Pacifique (Mumbai) (ap-south-1)
- Région Asie-Pacifique (Osaka) (ap-northeast-3)
- Région Asie-Pacifique (Séoul) (ap-northeast-2)
- Région Asie-Pacifique (Singapour) (ap-southeast-1)
- Région Asie-Pacifique (Sydney) (ap-southeast-2)
- Région Asie-Pacifique (Tokyo) (ap-northeast-1)

- Région Canada (Centre) (ca-central-1)
- Région Canada Ouest (Calgary) (ca-west-1)
- Région Chine (Beijing) (cn-north-1)
- Région Chine (Ningxia) (cn-northwest-1)
- Région Europe (Francfort) (eu-central-1)
- Région Europe (Irlande) (eu-west-1)
- Région Europe (Londres) (eu-west-2)
- Région Europe (Milan) (eu-south-1)
- Région Europe (Paris) (eu-west-3)
- Région Europe (Espagne) (eu-south-2)
- Région Europe (Stockholm) (eu-north-1)
- Région Europe (Zurich) (eu-central-2)
- Région Israël (Tel Aviv) (il-central-1)
- Région Moyen-Orient (Bahreïn) (me-south-1)
- Région Moyen-Orient (Émirats arabes unis) (me-central-1)
- Région Amérique du Sud (Sao Paulo) (sa-east-1)

Rubriques

- Consulter les recommandations d'Amazon Redshift Advisor
- Recommandations Amazon Redshift Advisor

Consulter les recommandations d'Amazon Redshift Advisor

Vous pouvez accéder aux recommandations d'Amazon Redshift Advisor à l'aide de la console Amazon Redshift, de l'API Amazon Redshift ou. AWS CLI Pour accéder aux recommandations, vous devez disposer d'une autorisation `redshift:ListRecommendations` associée à votre rôle ou à votre identité IAM.

Consulter les recommandations d'Amazon Redshift Advisor sur la console provisionnée Amazon Redshift

Vous pouvez consulter les recommandations d'Amazon Redshift Advisor sur le. AWS Management Console

Pour consulter les recommandations d'Amazon Redshift Advisor relatives aux clusters Amazon Redshift sur la console

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Advisor.
3. Développez chaque recommandations pour voir des informations supplémentaires. Sur cette page, vous pouvez trier et regrouper les recommandations.

Afficher les recommandations d'Amazon Redshift Advisor à l'aide des opérations de l'API Amazon Redshift

Vous pouvez répertorier les recommandations d'Amazon Redshift Advisor pour les clusters Amazon Redshift à l'aide de l'API Amazon Redshift. Généralement, vous développez une application dans le langage de programmation de votre choix pour appeler l'`redshift:ListRecommendations`API à l'aide d'un AWS SDK. Pour plus d'informations, consultez ListRecommendationsle manuel Amazon Redshift API Reference.

Afficher les recommandations d'Amazon Redshift Advisor à l'aide des opérations AWS Command Line Interface

Vous pouvez répertorier les recommandations d'Amazon Redshift Advisor pour les clusters Amazon Redshift à l'aide du. AWS Command Line Interface Pour plus d'informations, consultez la section list-recommendations dans le manuel de référence des AWS CLI commandes.

Recommandations Amazon Redshift Advisor

Amazon Redshift Advisor met à votre disposition des recommandations sur la façon d'optimiser votre cluster Amazon Redshift afin d'augmenter ses performances et de réduire les coûts d'exploitation. Vous trouverez des explications sur chaque recommandation dans la console, comme décrit précédemment. Vous trouverez des informations supplémentaires sur ces recommandations dans les sections suivantes :

Rubriques

- Compresser les objets de fichiers Amazon S3 chargés par COPY
- Isolation de plusieurs bases de données actives
- Réallocation de la mémoire de gestion de charge de travail (WLM)

- Omission de l'analyse de compression durant l'exécution de la commande COPY
- Fractionnement des objets Amazon S3 chargés par COPY
- Mise à jour des statistiques de table
- Activer l’accélération des requêtes courtes
- Modifier les clés de distribution sur les tables
- Modification des clés de tri au niveau des tables
- Modifier les encodages de compression sur les colonnes
- Recommandations sur les types de données

Compresser les objets de fichiers Amazon S3 chargés par COPY

La commande COPY tire parti de l'architecture de traitement massivement parallèle (MPP) d'Amazon Redshift pour lire et charger des données en parallèle. Il peut lire des fichiers à partir d'Amazon S3, des tables DynamoDB et des sorties texte à partir d'un ou plusieurs hôtes distants.

En cas de chargement de grandes quantités de données, nous vous recommandons vivement d'utiliser la commande COPY pour charger des fichiers de données compressés à partir de S3. La compression de jeux de données volumineux permet de réduire le temps de chargement des fichiers sur Amazon S3. La commande COPY peut également accélérer le processus de chargement en décompressant les fichiers au fur et à mesure de leur lecture.

Analyse

Les performances des commandes COPY à exécution longue qui chargent de grands ensembles de données non compressés peuvent souvent être améliorées. L'analyse Advisor identifie les commandes COPY qui chargent de grands ensembles de données non compressés. Dans ce cas, Advisor génère une recommandation visant à mettre en œuvre la compression sur les fichiers sources dans Amazon S3.

Recommandation

Veillez à ce que chaque commande COPY qui charge une quantité de données significative ou qui s'exécute longuement ingère des objets de données compressés à partir d'Amazon S3. Vous pouvez identifier les commandes COPY qui chargent des jeux de données volumineux non compressés à partir d'Amazon S3 en exécutant la commande SQL suivante en tant que super-utilisateur.

```
SELECT
```

```
    wq.userid, query, exec_start_time AS starttime, COUNT(*) num_files,
    ROUND(MAX(wq.total_exec_time/1000000.0),2) execution_secs,
    ROUND(SUM(transfer_size)/(1024.0*1024.0),2) total_mb,
    SUBSTRING(querytxt,1,60) copy_sql
FROM stl_s3client s
JOIN stl_query q USING (query)
JOIN stl_wlm_query wq USING (query)
WHERE s.userid>1 AND http_method = 'GET'
    AND POSITION('COPY ANALYZE' IN querytxt) = 0
    AND aborted = 0 AND final_state='Completed'
GROUP BY 1, 2, 3, 7
HAVING SUM(transfer_size) = SUM(data_size)
AND SUM(transfer_size)/(1024*1024) >= 5
ORDER BY 6 DESC, 5 DESC;
```

Si les données temporaires restent dans Amazon S3 après leur chargement, ce qui est courant dans les architectures de lac de données, leur stockage sous une forme compressée peut réduire vos coûts de stockage.

Conseils d'implémentation

- La taille idéale des objets est de 1 à 128 Mo après compression.
- Vous pouvez compresser vos fichiers à l'aide des formats gzip, lzop ou bzip2.

Isolation de plusieurs bases de données actives

Comme bonne pratique, nous recommandons d'isoler les bases de données dans Amazon Redshift les unes des autres. Les requêtes s'exécutent dans une base de données spécifique et ne peuvent accéder aux données des autres bases de données du cluster. Toutefois, les requêtes que vous exécutez dans toutes les bases de données d'un cluster partagent le même espace de stockage de cluster sous-jacent et les mêmes ressources de calcul. Lorsqu'un cluster unique contient plusieurs bases de données actives, leurs charges de travail sont généralement sans lien.

Analyse

L'analyse Advisor passe en revue toutes les bases de données du cluster à la recherche des charges de travail actives s'exécutant en même temps. Si des charges de travail actives sont exécutées en même temps, Advisor génère une recommandation pour envisager la migration des bases de données vers des clusters Amazon Redshift distincts.

Recommandation

Vous devez envisager de déplacer chaque base de données faisant l'objet de requêtes actives vers un cluster dédié distinct. L'utilisation d'un cluster distinct peut réduire les conflits de ressources et améliorer les performances des requêtes. En effet, cela vous permet de définir la taille de chaque cluster en fonction du stockage, des coûts et des performances attendus pour chaque charge de travail. En outre, les charges de travail non liées utilisent souvent des configurations de gestion de charge de travail différentes.

Pour identifier les bases de données qui sont activement utilisées, vous pouvez utiliser la commande SQL suivante en tant que super-utilisateur :

```
SELECT database,
  COUNT(*) as num_queries,
  AVG(DATEDIFF(sec,starttime,endtime)) avg_duration,
  MIN(starttime) as oldest_ts,
  MAX(endtime) as latest_ts
FROM stl_query
WHERE userid > 1
GROUP BY database;
```

Conseils d'implémentation

- Étant donné qu'un utilisateur doit se connecter à chaque base de données de façon spécifique et que les requêtes ne peuvent accéder qu'à une seule base de données, le déplacement des bases de données vers d'autres clusters a un impact minimal sur les utilisateurs.
- Pour déplacer une base de données, vous pouvez procéder comme suit :
  1. Restaurez temporairement un instantané du cluster actuel sur un cluster de même taille.
  2. Supprimez toutes les bases de données du nouveau cluster, à l'exception de la base de données cible à déplacer.
  3. Redimensionnez le cluster sur un type de nœud approprié et tenez compte de la charge de travail de la base de données.

Réallocation de la mémoire de gestion de charge de travail (WLM)

Amazon Redshift achemine les requêtes utilisateur vers Implémentation de la gestion manuelle de la charge de travail en vue de leur traitement. La gestion de la charge de travail (WLM) définit comment

ces requêtes sont acheminées vers les files d'attente. Amazon Redshift alloue à chaque file d'attente une partie de la mémoire disponible du cluster. La mémoire d'une file d'attente est répartie entre les emplacements de requête de la file d'attente.

Lorsqu'une file d'attente est configurée avec plus d'emplacements que la charge de travail n'en requiert, la mémoire allouée à ces emplacements inutilisés est sous-utilisée. La réduction des emplacements configurés en fonction des exigences de la charge de travail maximale permet de redistribuer la mémoire sous-utilisée aux emplacements actifs et peut améliorer ainsi les performances des requêtes.

Analyse

L'analyse Advisor passe en revue les exigences en matière de simultanéité de la charge de travail afin d'identifier les files d'attente de requêtes ayant des emplacement inutilisés. Advisor génère une recommandation conseillant de réduire le nombre d'emplacements dans une file d'attente lorsqu'il détecte les éléments suivants :

- Une file d'attente avec des emplacements complètement inactifs pendant l'analyse.
- Une file d'attente avec plus de quatre emplacements dont au moins deux sont inactifs pendant l'analyse.

Recommandation

La réduction des emplacements configurés en fonction des exigences de la charge de travail maximale permet de redistribuer la mémoire sous-utilisée aux emplacements actifs. Envisagez de réduire le nombre d'emplacements configurés pour les files d'attente où les emplacements n'ont jamais été totalement utilisés. Pour identifier ces files d'attente, vous pouvez comparer les exigences horaires maximales des emplacements de chaque file d'attente en exécutant la commande SQL suivante en tant que super-utilisateur :

```
WITH
 generate_dt_series AS (select sysdate - (n * interval '5 second') as dt from (select
 row_number() over () as n from stl_scan limit 17280)),
 apex AS (
     SELECT iq.dt, iq.service_class, iq.num_query_tasks, count(iq.slot_count) as
 service_class_queries, sum(iq.slot_count) as service_class_slots
     FROM
         (select gds.dt, wq.service_class, wscc.num_query_tasks, wq.slot_count
         FROM stl_wlm_query wq
```

```
        JOIN stv_wlm_service_class_config wscc ON (wscc.service_class =
 wq.service_class AND wscc.service_class > 5)
        JOIN generate_dt_series gds ON (wq.service_class_start_time <= gds.dt AND
 wq.service_class_end_time > gds.dt)
        WHERE wq.userid > 1 AND wq.service_class > 5) iq
    GROUP BY iq.dt, iq.service_class, iq.num_query_tasks),
    maxes as (SELECT apex.service_class, trunc(apex.dt) as d, date_part(h,apex.dt) as
 dt_h, max(service_class_slots) max_service_class_slots
                    from apex group by apex.service_class, apex.dt,
 date_part(h,apex.dt))
SELECT apex.service_class - 5 AS queue, apex.service_class, apex.num_query_tasks AS
 max_wlm_concurrency, maxes.d AS day, maxes.dt_h || ':00 - ' || maxes.dt_h || ':59' as
 hour, MAX(apex.service_class_slots) as max_service_class_slots
FROM apex
JOIN maxes ON (apex.service_class = maxes.service_class AND apex.service_class_slots =
 maxes.max_service_class_slots)
GROUP BY  apex.service_class, apex.num_query_tasks, maxes.d, maxes.dt_h
ORDER BY apex.service_class, maxes.d, maxes.dt_h;
```

La colonne `max_service_class_slots` représente le nombre maximum d'emplacements de requêtes WLM dans la file d'attente de requêtes pour cette heure. S'il existe des files d'attente sous-utilisées, mettez en œuvre l'optimisation de la réduction des créneaux en modifiant un groupe de paramètres, comme décrit dans le Guide de gestion Amazon Redshift.

Conseils d'implémentation

- Si votre charge de travail est fortement variable en volume, veillez à ce que l'analyse capture une période d'utilisation maximale. Si tel n'a pas été le cas, exécutez la commande SQL précédente à plusieurs reprises afin de surveiller les exigences maximales en matière de simultanéité.
- Pour plus de détails sur l'interprétation des résultats des requêtes à partir du code SQL précédent, consultez le script wlm_apex_hourly.sql sur GitHub.

Omission de l'analyse de compression durant l'exécution de la commande COPY

Lorsque vous chargez des données dans une table vide avec un encodage de compression déclaré avec la commande COPY, Amazon Redshift applique une compression de stockage. Cette optimisation permet de s'assurer que les données de votre cluster sont stockées efficacement, même lorsqu'elles sont chargées par les utilisateurs finaux. L'analyse requise pour appliquer la compression peut prendre un certain temps.

Analyse

L'analyse Advisor recherche les opérations COPY qui ont été retardées par l'analyse de compression automatique. L'analyse détermine les encodages de compression en échantillonnant les données pendant leur chargement. Cet échantillon est similaire à celui effectué par la commande ANALYZE COMPRESSION.

Lorsque vous chargez des données dans le cadre d'un processus structuré, tel qu'un traitement par lot ETL (extraction, transformation, chargement) pendant la nuit, vous pouvez définir la compression à l'avance. Vous pouvez également optimiser vos définitions de table en omettant cette phase de façon permanente sans aucun impact négatif.

Recommandation

Pour améliorer la réactivité de la commande COPY en omettant la phase d'analyse de compression, implémentez l'une des actions suivantes :

- Utilisez le paramètre de colonne `ENCODE` lors de la création des tables que vous chargez à l'aide de la commande COPY.
- Désactivez totalement la compression en indiquant le paramètre `COMPUPDATE OFF` dans la commande COPY.

La meilleure solution est généralement d'utiliser l'encodage de colonne lors de la création de table, car cette approche permet également de conserver le bénéfice du stockage des données compressées sur le disque. Vous pouvez utiliser la commande ANALYZE COMPRESSION pour suggérer les encodages de compression, mais vous devez recréer la table pour appliquer ces encodages. Pour automatiser ce processus, vous pouvez utiliser le AWSColumnEncodingUtility, disponible sur GitHub.

Pour identifier les opérations COPY récentes qui ont déclenché une analyse de compression automatique, utilisez la commande SQL suivante :

```
  WITH xids AS (
    SELECT xid FROM stl_query WHERE userid>1 AND aborted=0
    AND querytxt = 'analyze compression phase 1' GROUP BY xid
    INTERSECT SELECT xid FROM stl_commit_stats WHERE node=-1)
SELECT a.userid, a.query, a.xid, a.starttime, b.complyze_sec,
    a.copy_sec, a.copy_sql
FROM (SELECT q.userid, q.query, q.xid, date_trunc('s',q.starttime)
```

```
    starttime, substring(querytxt,1,100) as copy_sql,
    ROUND(datediff(ms,starttime,endtime)::numeric / 1000.0, 2) copy_sec
    FROM stl_query q JOIN xids USING (xid)
    WHERE (querytxt ilike 'copy %from%' OR querytxt ilike '% copy %from%')
    AND querytxt not like 'COPY ANALYZE %') a
LEFT JOIN (SELECT xid,
    ROUND(sum(datediff(ms,starttime,endtime))::numeric / 1000.0,2) complyze_sec
    FROM stl_query q JOIN xids USING (xid)
    WHERE (querytxt like 'COPY ANALYZE %'
    OR querytxt like 'analyze compression phase %')
    GROUP BY xid ) b ON a.xid = b.xid
WHERE b.complyze_sec IS NOT NULL ORDER BY a.copy_sql, a.starttime;
```

Conseils d'implémentation

- Veillez à ce que toutes les tables de taille significative créées durant vos processus ETL (par exemple, tables intermédiaires et tables temporaires) déclarent un encodage de compression pour toutes les colonnes, exceptée la première clé de tri.
- Estimez la taille de la table en cours de chargement pour toute sa durée de vie pour chacune des commandes COPY identifiées par la commande SQL précédente. Si vous êtes sûr que la table restera extrêmement petite, désactivez totalement la compression à l'aide du paramètre `COMPUPDATE OFF`. Sinon, créez la table avec une compression explicite avant de la charger avec la commande COPY.

Fractionnement des objets Amazon S3 chargés par COPY

La commande COPY tire parti de l'architecture de traitement massivement parallèle (MPP) d'Amazon Redshift pour lire et charger des données à partir de fichiers sur Amazon S3. La commande COPY charge les données en parallèle à partir de plusieurs fichiers, répartissant la charge de travail entre les nœuds de votre cluster. Afin d'obtenir un débit optimal, nous vous recommandons vivement de fractionner vos données en plusieurs fichiers pour tirer parti du traitement parallèle.

Analyse

L'analyse Advisor identifie les commandes COPY qui chargent les jeux de données volumineux contenus dans un petit nombre de fichiers placés temporairement dans Amazon S3. Les performances des commandes COPY à exécution longue qui chargent de grands ensembles de données à partir de quelques fichiers peuvent souvent être améliorées. Quand Advisor détermine que l'exécution de ces commandes COPY prend un temps significatif, il crée une recommandation

conseillant d'accroître le parallélisme en fractionnant les données en fichiers supplémentaires dans Amazon S3.

Recommandation

Dans ce cas, nous recommandons les actions suivantes, répertoriées par ordre de priorité :

1. Optimisez les commandes COPY qui chargent moins de fichiers que le nombre de nœuds de cluster.
2. Optimisez les commandes COPY qui chargent moins de fichiers que le nombre de tranches de cluster.
3. Optimisez les commandes COPY où le nombre de fichiers n'est pas un multiple du nombre de tranches du cluster.

Certaines commandes COPY chargent une quantité importante de données ou ont une durée d'exécution importante. Pour ces commandes, nous vous recommandons de charger un nombre d'objets de données d'Amazon S3 équivalent à un multiple du nombre de tranches présentes dans le cluster. Pour identifier combien d'objets S3 chaque commande COPY a chargé, exécutez le code SQL suivant en tant que super-utilisateur.

```
SELECT
    query, COUNT(*) num_files,
    ROUND(MAX(wq.total_exec_time/1000000.0),2) execution_secs,
    ROUND(SUM(transfer_size)/(1024.0*1024.0),2) total_mb,
    SUBSTRING(querytxt,1,60) copy_sql
FROM stl_s3client s
JOIN stl_query q USING (query)
JOIN stl_wlm_query wq USING (query)
WHERE s.userid>1 AND http_method = 'GET'
    AND POSITION('COPY ANALYZE' IN querytxt) = 0
    AND aborted = 0 AND final_state='Completed'
GROUP BY query, querytxt
HAVING (SUM(transfer_size)/(1024*1024))/COUNT(*) >= 2
ORDER BY CASE
WHEN COUNT(*) < (SELECT max(node)+1 FROM stv_slices) THEN 1
WHEN COUNT(*) < (SELECT COUNT(*) FROM stv_slices WHERE node=0) THEN 2
ELSE 2+((COUNT(*) % (SELECT COUNT(*) FROM stv_slices))/(SELECT COUNT(*)::DECIMAL FROM
 stv_slices))
END, (SUM(transfer_size)/(1024.0*1024.0))/COUNT(*) DESC;
```

Conseils d'implémentation

- Le nombre de tranches par nœud dépend de la taille de nœud du cluster. Pour plus d'informations sur le nombre de tranches dans les différents types de nœuds, consultez la rubrique Clusters et nœuds dans Amazon Redshift dans le Guide de gestion Amazon Redshift.
- Vous pouvez charger plusieurs fichiers en spécifiant un préfixe commun, ou clé de préfixe, pour l'ensemble, ou en indiquant explicitement les fichiers dans un fichier manifeste. Pour plus d'informations sur le chargement des fichiers, consultez Chargement de données à partir de fichiers compressés et non compressés.
- Amazon Redshift ne prend pas en compte la taille des fichiers lors de la répartition de la charge de travail. Fractionnez vos fichiers de données de chargement de telle sorte que les fichiers soient à peu près de taille égale, entre 1 Mo et 1 Go après compression.

Mise à jour des statistiques de table

Amazon Redshift utilise un optimiseur de requêtes basé sur les coûts pour choisir le plan d'exécution optimal des requêtes. Les estimations de coût reposent sur les statistiques de table générées via la commande ANALYZE. Lorsque les statistiques sont obsolètes ou manquantes, la base de données peut choisir un plan d'exécution des requêtes moins efficace, en particulier pour les requêtes complexes. Le maintien à jour des statistiques permet d'exécuter des requêtes complexes en un temps minimal.

Analyse

L'analyse Advisor permet de suivre les tables dont les statistiques sont out-of-date ou sont absentes. Elle passe en revue les métadonnées d'accès aux tables qui sont associées à des requêtes complexes. Si des statistiques sont manquantes pour des tables qui font l'objet d'accès fréquents avec des modèles complexes, Advisor crée une recommandation critique conseillant d'exécuter la commande ANALYZE. Si les tables fréquemment consultées avec des modèles complexes contiennent des out-of-date statistiques, Advisor propose une recommandation pour exécuter ANALYZE.

Recommandation

Chaque fois que le contenu de la table change de façon significative, mettez à jour les statistiques à l'aide de la commande ANALYZE. Nous recommandons d'exécuter la commande ANALYZE chaque

fois qu'un nombre important de nouvelles lignes de données est chargé dans une table existante à l'aide de la commande COPY ou INSERT. Nous recommandons également d'exécuter la commande ANALYZE chaque fois qu'un nombre important de lignes sont modifiées à l'aide de la commande UPDATE ou DELETE. Pour identifier les tables manquantes ou out-of-date les statistiques, exécutez la commande SQL suivante en tant que superutilisateur. Les résultats sont classés de la plus grande à la plus petite table.

Pour identifier les tables manquantes ou out-of-date les statistiques, exécutez la commande SQL suivante en tant que superutilisateur. Les résultats sont classés de la plus grande à la plus petite table.

```
SELECT
   ti.schema||'.'||ti."table" tablename,
   ti.size table_size_mb,
   ti.stats_off statistics_accuracy
 FROM svv_table_info ti
 WHERE ti.stats_off > 5.00
 ORDER BY ti.size DESC;
```

Conseils d'implémentation

Le seuil par défaut d'ANALYZE est de 10 %. Cela signifie que la commande ANALYZE ignore une table donnée si moins de 10 % des lignes de la table ont été modifiées depuis la dernière exécution d'ANALYZE. Par conséquent, vous pouvez choisir d'émettre les commandes ANALYZE à la fin de chaque processus ETL. Cette approche signifie que la commande ANALYZE est souvent ignorée mais vous permet d'être sûr qu'elle n'est exécutée que lorsque cela est nécessaire.

C'est sur les colonnes utilisées dans les jointures (par exemple, `JOIN tbl_a ON col_b`) ou comme prédicats (par exemple, `WHERE col_b = 'xyz'`) que les statistiques ANALYZE ont le plus fort impact. Par défaut, la commande ANALYZE collecte des statistiques pour toutes les colonnes de la table spécifiée. Si nécessaire, vous pouvez réduire le temps requis pour exécuter la commande ANALYZE en ne l'exécutant que pour les colonnes sur lesquelles elle a le plus d'impact. Vous pouvez exécuter la commande SQL suivante pour identifier les colonnes utilisées comme prédicats : Vous pouvez également laisser Amazon Redshift choisir les colonnes à analyser en spécifiant `ANALYZE PREDICATE COLUMNS`.

```
WITH predicate_column_info as (
```

```
SELECT ns.nspname AS schema_name, c.relname AS table_name, a.attnum as col_num,
 a.attname as col_name,
        CASE
            WHEN 10002 = s.stakind1 THEN array_to_string(stavalues1, '||')
            WHEN 10002 = s.stakind2 THEN array_to_string(stavalues2, '||')
            WHEN 10002 = s.stakind3 THEN array_to_string(stavalues3, '||')
            WHEN 10002 = s.stakind4 THEN array_to_string(stavalues4, '||')
            ELSE NULL::varchar
        END AS pred_ts
   FROM pg_statistic s
   JOIN pg_class c ON c.oid = s.starelid
   JOIN pg_namespace ns ON c.relnamespace = ns.oid
   JOIN pg_attribute a ON c.oid = a.attrelid AND a.attnum = s.staattnum)
SELECT schema_name, table_name, col_num, col_name,
       pred_ts NOT LIKE '2000-01-01%' AS is_predicate,
       CASE WHEN pred_ts NOT LIKE '2000-01-01%' THEN (split_part(pred_ts,
 '||',1))::timestamp ELSE NULL::timestamp END as first_predicate_use,
       CASE WHEN pred_ts NOT LIKE '%||2000-01-01%' THEN (split_part(pred_ts,
 '||',2))::timestamp ELSE NULL::timestamp END as last_analyze
FROM predicate_column_info;
```

Pour plus d'informations, consultez Analyse des tables.

Activer l’accélération des requêtes courtes

L’accélération des requêtes courtes (SQA) établit la priorité des requêtes de courte durée sélectionnées sur les requêtes de longue durée. SQA exécute des requêtes de courte durée dans un espace dédié, afin que les requêtes SQA ne soient pas forcées d’attendre dans des files d’attente derrière les requêtes de longue durée. SQA priorise uniquement les requêtes de courte durée qui sont placées dans une file d’attente définie par l’utilisateur. Avec SQA, les requêtes de courte durée commencent à s’exécuter plus rapidement et les utilisateurs obtiennent les résultats plus rapidement.

Si vous activez l'accélération des requêtes courtes (SQA), vous pouvez réduire ou éliminer les files d'attente de gestion de charge de travail (WLM) dédiées à l'exécution des requêtes courtes. De plus, les requêtes de longue durée n’ont pas besoin de se heurter à des requêtes courtes pour obtenir des emplacements dans une file d’attente afin que vous puissiez configurer vos files d’attente WLM pour utiliser moins d’emplacements de requêtes. Lorsque vous utilisez une simultanéité réduite, le débit de requêtes est accru et les performances systèmes globales sont améliorées pour la plupart des charges de travail. Pour plus d'informations, consultez Accélération des requêtes courtes.

Analyse

Advisor vérifie les modèles de charge de travail et indique le nombre de requêtes récentes où l'accélération des requêtes courtes doit réduire la latence et le temps d'attente quotidien pour les requêtes admissibles à l'accélération des requêtes courtes.

Recommandation

Modifiez la configuration de WLM pour activer l'accélération SQA. Amazon Redshift utilise un algorithme de machine learning pour analyser chaque requête éligible. Les prédictions s'améliorent à mesure que l'accélération des requêtes courtes apprend de vos modèles de requêtes. Pour plus d'informations, consultez Configuration de la gestion de la charge de travail.

Lorsque vous activez l'accélération SQA, WLM définit par défaut le temps d'exécution maximal des requêtes courtes sur « dynamic ». Il est recommandé de conserver cette valeur pour la durée d'exécution maximale SQA.

Conseils d'implémentation

Pour vérifier si l'accélération SQA est activée, exécutez la requête suivante. Si la requête renvoie une ligne, l'accélération SQA est activée.

```
select * from stv_wlm_service_class_config
where service_class = 14;
```

Pour de plus amples informations, veuillez consulter Surveillance de la SQA.

Modifier les clés de distribution sur les tables

Amazon Redshift distribue les lignes de la table dans le cluster selon le style de distribution de la table. Les tables avec une distribution KEY nécessitent une colonne comme clé de distribution (DISTKEY). Une ligne de table est affectée à une tranche de nœud d'un cluster en fonction de la valeur de sa colonne DISTKEY.

Une colonne DISTKEY appropriée place un nombre similaire de lignes sur chaque tranche de nœud et est fréquemment référencée dans les conditions de jonction. Une jonction optimisée se produit lorsque les tables sont jointes sur leurs colonnes DISTKEY, accélérant ainsi la performance des requêtes.

Analyse

Advisor analyse la charge de travail de votre cluster pour identifier la clé de distribution la plus appropriée aux tables pouvant bénéficier de manière significative d'un style de distribution KEY.

Recommandation

Advisor fournit des instructions ALTER TABLE qui modifient les colonnes DISTSTYLE et DISTKEY d'une table en fonction de son analyse. Pour obtenir un avantage significatif en termes de performances, toutes les instructions SQL d'un groupe de recommandations doivent être implémentées.

La redistribution d'une table de taille importante avec ALTER TABLE consomme des ressources de cluster et nécessite des verrouillages temporaires de table à différents moments. Implémentez chaque groupe de recommandations lorsque les autres charges de travail du cluster sont légères. Pour plus d'informations sur l'optimisation des propriétés d'une distribution de table, consultez l'article Amazon Redshift Engineering's Advanced Table Design Playbook : Distribution Styles and Distribution Keys.

Pour plus d'informations sur ALTER DISTSYLE et DISTKEY, consultez ALTER TABLE.

> **Note**
>
> Si vous ne voyez pas de recommandation, cela ne signifie pas nécessairement que les styles de distribution actuels sont les plus appropriés. Advisor s'abstient de formuler des recommandations lorsqu'il n'y a pas suffisamment de données ou que les avantages escomptés de la redistribution sont faibles.
> Les recommandations d'Advisor s'appliquent à une table particulière et ne s'appliquent pas nécessairement à une table contenant une colonne portant le même nom. Les tables qui partagent un nom de colonne peuvent avoir des caractéristiques différentes, à moins que les données à l'intérieur des tables soient les mêmes.
> Si vous voyez des recommandations pour des tables intermédiaires créées ou abandonnées par des tâches ETL, modifiez vos processus ETL pour utiliser les clés de distribution recommandées par Advisor.

Modification des clés de tri au niveau des tables

Amazon Redshift trie les lignes des tables en fonction de leur clé de tri. Le tri des lignes de table repose sur les valeurs des colonnes de clé de tri.

Le tri d'une table en fonction d'une clé de tri appropriée peut accélérer les performances des requêtes, en particulier celles dont les prédicats sont limités à la plage, en réduisant le nombre de blocs de table à lire à partir du disque.

Analyse

Advisor analyse la charge de travail de votre cluster sur plusieurs jours afin d'identifier une clé de tri utile pour vos tables.

Recommandation

Advisor fournit deux groupes d'instructions ALTER TABLE qui modifient la clé de tri d'une table en fonction de son analyse :

- Instructions qui modifient une table qui ne possède actuellement pas de clé de tri pour ajouter une clé de tri COMPOUND.
- Instructions qui modifient une clé de tri depuis INTERLEAVED en COMPOUND ou en aucune clé de tri.

  L'utilisation de clés de tri composé réduit de façon importante la charge de maintenance. Les tables dotées de clés de tri composé n'ont pas besoin des opérations VACUUM REINDEX onéreuses qui sont requises pour les tris entrelacés. En pratique, les clés de tri composées sont plus efficaces que les clés de tri entrelacées pour la grande majorité des charges de travail d'Amazon Redshift. Toutefois, si une table est petite, il est plus efficace de ne pas avoir de clé de tri pour éviter les frais généraux de stockage des clés de tri.

Lors du tri d'une table volumineuse avec ALTER TABLE, les ressources de cluster sont consommées et des verrous de table sont requis à différents moments. Mettez en œuvre chaque recommandation lorsque la charge de travail d'un cluster est modérée. Pour plus d'informations sur l'optimisation des configurations des clés de tri de table, consultez l'article Amazon Redshift Engineering's Advanced Table Design Playbook: Compound and Interleaved Sort Keys.

Pour plus d'informations sur ALTER SORTKEY, consultez ALTER TABLE.

> **Note**
>
> Si vous ne voyez pas de recommandation pour une table, cela ne signifie pas nécessairement que la configuration actuelle est la plus appropriée. Advisor s'abstient de formuler des recommandations lorsqu'il n'y a pas suffisamment de données ou que les avantages escomptés du tri sont faibles.
> Les recommandations d'Advisor s'appliquent à une table particulière et pas nécessairement à une table contenant une colonne présentant les mêmes nom et type de données. Les tables

qui partagent des noms de colonne peuvent faire l'objet de recommandations différentes en fonction de leurs données et de la charge de travail.

Modifier les encodages de compression sur les colonnes

La compression est une opération au niveau des colonnes qui réduit la taille des données lorsque celles-ci sont stockées. La compression est utilisée dans Amazon Redshift pour conserver l'espace de stockage et améliorer les performances des requêtes en réduisant la quantité des I/O sur le disque. Nous recommandons un encodage de compression optimal pour chaque colonne en fonction de son type de données et des modèles de requête. Avec une compression optimale, les requêtes peuvent s'exécuter plus efficacement et la base de données peut occuper un espace de stockage minimal.

Analyse

Advisor analyse en permanence la charge de travail de votre cluster et le schéma de la base de données afin d'identifier l'encodage de compression optimal pour chaque colonne de la table.

Recommandation

Advisor fournit des instructions ALTER TABLE qui modifient l'encodage de compression de colonnes particulières, sur la base de son analyse.

Le changement des encodages de compression des colonnes avec ALTER TABLE consomme des ressources du cluster et nécessite des verrous de table à différents moments. Il est préférable de mettre en œuvre les recommandations lorsque la charge de travail du cluster est légère.

Pour référence, Exemples ALTER TABLE présente plusieurs instructions qui modifient l'encodage d'une colonne.

> **Note**
>
> Advisor s'abstient de formuler des recommandations lorsqu'il n'y a pas assez de données ou que le bénéfice attendu d'un changement de codage est faible.

Recommandations sur les types de données

Amazon Redshift dispose d'une bibliothèque de types de données SQL pour différents cas d'utilisation. Il s'agit notamment de types entiers comme `INT` et de types destinés à stocker des

personnages, comme VARCHAR. Redshift stocke les types de manière optimisée afin de fournir un accès rapide et de bonnes performances de requête. De plus, Redshift fournit des fonctions pour des types spécifiques, que vous pouvez utiliser pour formater ou effectuer des calculs sur des résultats de requête.

Analyse

Advisor analyse en permanence la charge de travail de votre cluster et le schéma de la base de données afin d'identifier les colonnes susceptibles de bénéficier considérablement d'un changement de type de données.

Recommandation

Advisor fournit une instruction ALTER TABLE qui ajoute une nouvelle colonne avec le type de données suggéré. Une instruction UPDATE d'accompagnement copie les données de la colonne existante vers la nouvelle colonne. Après avoir créé la nouvelle colonne et chargé les données, modifiez vos requêtes et vos scripts d'ingestion pour accéder à la nouvelle colonne. Exploitez ensuite les fonctionnalités et fonctions spécialisées dans le nouveau type de données, que l'on trouve dans Référence sur les fonctions SQL.

La copie de données existantes dans la nouvelle colonne peut prendre du temps. Nous vous recommandons d'implémenter chaque recommandation d'Advisor lorsque la charge de travail du cluster est légère. Consultez la liste des types de données disponibles à l'adresse Types de données.

Notez qu'Advisor s'abstient de formuler des recommandations lorsqu'il n'y a pas assez de données ou que le bénéfice attendu d'un changement de type de données est faible.

Didacticiels pour Amazon Redshift

Suivez les étapes décrites dans ces didacticiels pour en savoir plus sur les fonctions Amazon Redshift :

Didacticiel : chargement des données à partir d'Amazon S3	Dans ce didacticiel, vous allez suivre le processus de chargement de données dans vos tables de base de données Amazon Redshift à partir de fichiers de données d'un compartiment S3 du début à la fin.
Didacticiel : Faire une requête de données imbriquées avec Amazon Redshift Spectrum	Dans ce didacticiel, vous allez utiliser Redshift Spectrum pour interroger des données imbriquées. Redshift Spectrum vous permet d'interroger des données dans les formats de fichier Parquet, ORC, JSON ou Ion.
Didacticiel : Configuration des files d'attente de gestion manuelle de la charge de travail	Dans ce didacticiel, vous allez configure r Amazon Redshift pour utiliser des files d'attente de gestion manuelle de la charge de travail (WLM). Amazon Redshift utilise des files d'attente WLM pour gérer la répartition des ressources afin d'exécuter des requêtes simultanées. Vous devez configurer Amazon Redshift pour utiliser le WLM manuel si vous devez utiliser plusieurs files d'attente WLM WLM.
Didacticiel : Utiliser les fonctions SQL spatiales avec Amazon Redshift	Dans ce didacticiel, vous allez interroger des données à l'aide de fonctions spatiales. Vous utilisez des fonctions spatiales pour interroger des données de géométrie et de géographie.
Tutoriels pour Amazon Redshift ML	Dans ces didacticiels, vous allez créer et utiliser des modèles d'apprentissage automatique.

Tutoriel : Création de rôles et interrogation avec RBAC	Dans ce didacticiel, vous allez utiliser le contrôle d'accès basé sur les rôles (RBAC) pour créer et utiliser des autorisations dans une base de données que vous créez. Avec RBAC, vous pouvez créer des rôles dotés d'autorisations spécifiques, telles que des autorisations en lecture seule ou en lecture-écriture. Vous pouvez ensuite attribuer ces rôles aux utilisateurs pour leur accorder les autorisations spécifiées.

Optimisation automatique des tables

L'optimisation automatique des tables est une fonction d'autoréglage qui optimise automatiquement la conception des tables en appliquant des clés de tri et de distribution sans intervention de l'administrateur. En utilisant l'automatisation pour optimiser la conception des tables, vous pouvez démarrer et obtenir les performances les plus rapides sans avoir à investir du temps pour ajuster et mettre en œuvre manuellement les optimisations des tables.

L'optimisation automatique des tables observe en permanence la façon dont les requêtes interagissent avec les tables. Elle utilise des méthodes d'intelligence artificielle avancées pour choisir les clés de tri et de distribution afin d'optimiser les performances pour la charge de travail du cluster. Si Amazon Redshift détermine que l'application d'une clé améliore les performances du cluster, les tables sont automatiquement modifiées dans les heures suivant la création du cluster, avec un impact minimal sur les requêtes.

Pour tirer parti de cette automatisation, un administrateur Amazon Redshift crée une nouvelle table ou modifie une table existante pour lui permettre d'utiliser l'optimisation automatique. Les tables existantes avec un style de distribution ou une clé de tri `AUTO` sont déjà activées pour l'automatisation. Lorsque vous exécutez des requêtes sur ces tables, Amazon Redshift détermine si une clé de tri ou une clé de distribution améliorera les performances. Si c'est le cas, Amazon Redshift modifie alors automatiquement la table sans nécessiter l'intervention de l'administrateur. Si un nombre minimum de requêtes est exécuté, les optimisations sont appliquées dans les heures suivant le lancement du cluster.

Si Amazon Redshift détermine qu'une clé de distribution améliore les performances des requêtes, les tables dont le style de distribution est `AUTO` peuvent voir celui-ci modifié en `KEY`.

Rubriques

- Activation, désactivation et surveillance de l'optimisation automatique des tables
- Compression de colonnes pour réduire la taille des données stockées
- Distribution des données pour l'optimisation des requêtes
- Clés de tri
- Contraintes liées aux tables

Activation, désactivation et surveillance de l'optimisation automatique des tables

Par défaut, les tables créées sans définir explicitement les clés de tri ou de distribution sont définies sur AUTO. Au moment de la création de la table, vous pouvez également définir explicitement un tri ou une clé de distribution manuellement. Si vous définissez la clé de tri ou de distribution, la table n'est pas gérée automatiquement.

Activation de l'optimisation automatique des tables

Pour permettre à une table existante d'être automatiquement optimisée, utilisez les options de l'instruction ALTER pour changer la table en AUTO. Vous pouvez choisir de définir l'automatisation pour les clés de tri, mais pas pour les clés de distribution (et vice versa). Si vous exécutez une instruction ALTER pour convertir une table en table automatisée, les clés de tri et les styles de distribution existants sont conservés.

```
ALTER TABLE table_name ALTER SORTKEY AUTO;
```

```
ALTER TABLE table_name ALTER DISTSTYLE AUTO;
```

Pour plus d'informations, consultez ALTER TABLE.

Initialement, une table n'a pas de clé de distribution ni de clé de tri. Le style de distribution est défini sur EVEN ou ALL selon la taille de la table. Au fur et à mesure que la taille de la table augmente, Amazon Redshift applique les clés de distribution et les clés de tri optimales. Les optimisations sont appliquées dans les heures qui suivent l'exécution d'un nombre minimum de requêtes. Lorsqu'il détermine les optimisations des clés de tri, Amazon Redshift tente d'optimiser les blocs de données lus sur le disque pendant l'analyse d'une table. Lors de la détermination des optimisations de style de distribution, Amazon Redshift tente d'optimiser le nombre d'octets transférés entre les nœuds de cluster.

Désactivation de l'optimisation automatique d'une table

Vous pouvez désactiver l'optimisation automatique d'une table. La désactivation de l'automatisation d'une table implique la sélection d'une clé de tri ou d'un style de distribution. Pour modifier le style de distribution, spécifiez un style de distribution spécifique.

```
ALTER TABLE table_name ALTER DISTSTYLE EVEN;
```

```
ALTER TABLE table_name ALTER DISTSTYLE ALL;
```

```
ALTER TABLE table_name ALTER DISTSTYLE KEY DISTKEY c1;
```

Pour modifier une clé de tri, vous pouvez définir une clé de tri ou n'en choisir aucune.

```
ALTER TABLE table_name ALTER SORTKEY(c1, c2);
```

```
ALTER TABLE table_name ALTER SORTKEY NONE;
```

Surveillance de l'optimisation automatique des tables

La vue système SVV_ALTER_TABLE_RECOMMENDATIONS enregistre les recommandations Amazon Redshift Advisor actuelles pour les tables. Cette vue affiche des recommandations pour toutes les tables, celles qui sont définies pour l'optimisation automatique et celles qui ne le sont pas.

Pour savoir si une table est définie pour l'optimisation automatique, interrogez la vue système SVV_TABLE_INFO. Les entrées ne s'affichent que pour les tables visibles dans la base de données de la séance en cours. Les recommandations sont insérées dans la vue deux fois par jour, dans les heures qui suivent la création du cluster. Dès qu'une recommandation est disponible, elle est appliquée dans l'heure qui suit. Une fois qu'une recommandation a été appliquée (soit par Amazon Redshift, soit par vous), elle n'apparaît plus dans la vue.

La vue système SVL_AUTO_WORKER_ACTION affiche un journal d'audit de toutes les actions effectuées par Amazon Redshift, ainsi que l'état précédent de la table.

La vue système SVV_TABLE_INFO répertorie toutes les tables du système, ainsi qu'une colonne pour indiquer si la clé de tri et le style de distribution de la table sont définis sur AUTO.

Pour plus d'informations sur ces vues système, consultez Surveillance système (clusters mis en service uniquement).

Compression de colonnes pour réduire la taille des données stockées

La compression est une opération au niveau des colonnes qui réduit la taille des données lorsque celles-ci sont stockées. La compression préserve l'espace de stockage et réduit la taille des données qui sont lues depuis le stockage, ce qui réduit la quantité d'I/O de disque et par conséquent améliore les performances des requêtes.

ENCODE AUTO est la valeur par défaut pour les tables. Lorsqu'une table est définie sur ENCODE AUTO, Amazon Redshift gère automatiquement l'encodage de compression pour toutes les colonnes de la table. Pour plus d'informations, consultez CREATE TABLE et ALTER TABLE.

Cependant, si vous spécifiez l'encodage de compression pour une colonne quelconque du tableau, le tableau n'est plus défini sur ENCODE AUTO. Amazon Redshift ne gère plus automatiquement le codage de compression pour toutes les colonnes de la table.

Vous pouvez appliquer un type de compression, aussi appelé encodage, aux colonnes d'une table manuellement lorsque vous créez la table. Vous pouvez également utiliser l'instruction COPY pour analyser et appliquer automatiquement la compression. Pour plus d'informations, consultez Laissez COPY choisir les encodages de compression. Pour plus de détails sur la mise en œuvre de la compression automatique, consultez Chargement des tables avec compression automatique.

> **Note**
>
> Nous vous recommandons vivement d'utiliser la commande COPY pour appliquer la compression automatique.

Vous pouvez choisir d'appliquer les encodages de compression manuellement si la nouvelle table partage les mêmes caractéristiques de données qu'une autre table. Vous pouvez également le faire si vous découvrez, lors des tests, que les encodages de compression appliqués lors de la compression automatique ne sont pas les mieux adaptés à vos données. Si vous choisissez d'appliquer manuellement des encodages de compression, vous pouvez exécuter la commande ANALYZE COMPRESSION dans une table déjà renseignée et utiliser les résultats pour choisir les encodages de compression.

Pour appliquer manuellement la compression, vous spécifiez les encodages de compression de chaque colonne dans le cadre de l'instruction CREATE TABLE. La syntaxe est la suivante.

```
CREATE TABLE table_name (column_name
data_type ENCODE encoding-type)[, ...]
```

Ici, le type d'encodage (encoding-type) est repris du tableau des mots-clés de la section suivante.

Par exemple, l'instruction suivante crée une table à deux colonnes, PRODUCT. Lorsque les données sont chargées dans la table, la colonne PRODUCT_ID n'est pas compressée, mais la colonne PRODUCT_NAME l'est à l'aide de l'encodage par dictionnaire d'octets (BYTEDICT).

```
create table product(
product_id int encode raw,
product_name char(20) encode bytedict);
```

Vous pouvez spécifier l'encodage d'une colonne lorsque celle-ci est ajoutée à une table à l'aide de la commande ALTER TABLE.

```
ALTER TABLE table-name ADD [ COLUMN ] column_name column_type ENCODE encoding-type
```

Rubriques

- encodages de compression
- Test des encodages de compression

encodages de compression

Un encodage de compression spécifie le type de compression appliqué à une colonne de valeurs de données à mesure que les lignes sont ajoutées à une table.

ENCODE AUTO est la valeur par défaut pour les tables. Lorsqu'une table est définie sur ENCODE AUTO, Amazon Redshift gère automatiquement l'encodage de compression pour toutes les colonnes de la table. Pour plus d'informations, consultez CREATE TABLE et ALTER TABLE.

Cependant, si vous spécifiez l'encodage de compression pour une colonne quelconque du tableau, le tableau n'est plus défini sur ENCODE AUTO. Amazon Redshift ne gère plus automatiquement le codage de compression pour toutes les colonnes de la table.

Lorsque vous utilisez CREATE TABLE, ENCODE AUTO est désactivé lorsque vous spécifiez un encodage de compression pour une colonne de la table. Si ENCODE AUTO est désactivé, Amazon

Redshift attribue automatiquement un encodage de compression aux colonnes pour lesquelles vous ne spécifiez pas de type ENCODE, comme suit :

- Les colonnes qui sont définies comme des clés de tri se voient attribuer une compression RAW.
- Les colonnes qui sont définies comme des types de données BOOLEAN, REAL ou DOUBLE PRECISION se voient attribuer une compression RAW.
- Les colonnes définies comme des types de données SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIMESTAMP ou TIMESTAMPTZ sont soumises à une compression. AZ64
- Les colonnes définies en tant que types de données CHAR ou VARCHAR sont affectées à une compression LZO.

Vous pouvez modifier l'encodage d'une table après l'avoir créée en utilisant ALTER TABLE.
Si vous désactivez ENCODE AUTO à l'aide de ALTER TABLE, Amazon Redshift ne gère plus automatiquement les encodages de compression pour vos colonnes. Toutes les colonnes conserveront les types d'encodage de compression qu'elles avaient lorsque vous avez désactivé ENCODE AUTO jusqu'à ce que vous les changiez ou que vous activiez à nouveau ENCODE AUTO.

Amazon Redshift prend en charge les codages de compression suivants :

Raw

L'encodage brut est l'encodage par défaut pour les colonnes désignées comme clés de tri et les colonnes définies avec le type de données BOOLEAN, REAL ou DOUBLE PRECISION. Avec l'encodage brut, les données sont stockées dans un format brut, non compressé.

AZ64

AZ64 est un algorithme de codage par compression propriétaire conçu par Amazon pour atteindre un taux de compression élevé et améliorer le traitement des requêtes. À la base, l' AZ64 algorithme compresse de petits groupes de valeurs de données et utilise des instructions SIMD (single instruction multiple data) pour le traitement en parallèle. AZ64À utiliser pour réaliser des économies de stockage importantes et des performances élevées pour les types de données numériques, de date et d'heure.

Vous pouvez l'utiliser AZ64 comme codage de compression lorsque vous définissez des colonnes à l'aide des instructions CREATE TABLE et ALTER TABLE avec les types de données suivants :

- SMALLINT
- INTEGER

- BIGINT
- DECIMAL
- DATE
- TIMESTAMP
- TIMESTAMPTZ

Byte-dictionary

Avec l'encodage par dictionnaire d'octets, un dictionnaire distinct de valeurs uniques est créé pour chaque bloc de valeurs de colonne sur le disque. (Un bloc de disque Amazon Redshift occupe 1 Mo.) Le dictionnaire contient jusqu'à 256 valeurs d'un octet qui sont stockées sous forme d'index pour les valeurs de données originales. Si plus de 256 valeurs sont stockées dans un seul bloc, les valeurs supplémentaires sont écrites dans le bloc dans un format brut, non compressé. Le processus se répète pour chaque bloc de disque.

Ce codage est très efficace pour les colonnes de chaînes à faible cardinalité. Cet encodage est optimal lorsque le domaine de données d'une colonne est inférieur à 256 valeurs uniques.

Pour les colonnes avec le type de données chaîne (CHAR et VARCHAR) codées avec BYTEDICT, Amazon Redshift effectue des analyses vectorisées et des évaluations de prédicats qui opèrent directement sur les données compressées. Ces analyses utilisent des instructions SIMD (Single Instruction and Multiple Data) spécifiques au matériel pour le traitement parallèle. Cela permet d'accélérer considérablement l'analyse des colonnes de chaînes de caractères. L'encodage par dictionnaire d'octets est particulièrement efficace en termes d'espace si une colonne CHAR/VARCHAR contient de longues chaînes de caractères.

Supposons qu'une table comporte une colonne COUNTRY dont le type de données est CHAR(30). A mesure que les données sont chargées, Amazon Redshift crée le dictionnaire et remplit la colonne COUNTRY avec la valeur de l'index. Le dictionnaire contient les valeurs uniques indexées, et la table elle-même contient uniquement les sous-scripts d'un octet des valeurs correspondantes.

> **Note**
>
> Les espaces de fin sont stockés pour les colonnes de caractères de longueur fixe. Par conséquent, dans une colonne CHAR(30), chaque valeur compressée enregistre 29 octets de stockage lorsque vous utilisez l'encodage par dictionnaire d'octets.

Le tableau suivant illustre le dictionnaire de la colonne COUNTRY.

Valeur de données unique	Index du dictionnaire	Taille (longueur fixe, 30 octets par valeur)
England	0	30
United States of America	1	30
Venezuela	2	30
Sri Lanka	3	30
Argentina	4	30
Japan	5	30
Total		180

Le tableau suivant représente les valeurs de la colonne COUNTRY.

Valeur de données d'origine	Taille d'origine (longueur fixe, 30 octets par valeur)	Valeur compressée (index)	Nouvelle taille (octets)
England	30	0	1
England	30	0	1
United States of America	30	1	1
United States of America	30	1	1
Venezuela	30	2	1
Sri Lanka	30	3	1

Valeur de données d'origine	Taille d'origine (longueur fixe, 30 octets par valeur)	Valeur compressée (index)	Nouvelle taille (octets)
Argentina	30	4	1
Japan	30	5	1
Sri Lanka	30	3	1
Argentina	30	4	1
Total	300		10

Dans cet exemple, la taille compressée totale est calculée comme suit : 6 entrées distinctes sont stockées dans le dictionnaire (6 * 30 = 180), et la table contient 10 valeurs compressées de 1 octet, soit un total de 190 octets.

Delta

Les encodages Delta sont très utiles pour les colonnes de date et d'heure.

L'encodage Delta compresse les données en enregistrant la différence entre les valeurs qui se suivent dans la colonne. Cette différence est enregistrée dans un dictionnaire distinct pour chaque bloc de valeurs de la colonne sur le disque. (Un bloc de disque Amazon Redshift occupe 1 Mo.) Par exemple, supposons que la colonne contienne 10 entiers dans l'ordre de 1 à 10. Les premiers sont stockés sous la forme d'un entier de 4 octets (plus un indicateur de 1 octet). Les neuf suivants sont chacun stockés sous forme d'octet avec la valeur 1, ce qui indique qu'il est supérieur à la valeur précédente.

L'encodage Delta se décline en deux variations :

- DELTA enregistre les différences sous forme de valeurs de 1 octets (nombres entiers de 8 bits)
- DELTA32K enregistre les différences sous forme de valeurs de 2 octets (entiers de 16 bits)

Si la plupart des valeurs de la colonne peuvent être comprimées en utilisant un seul octet, la variation d'un octet est très efficace. Cependant, si les deltas sont plus importants, cet encodage, dans le pire des cas, est un peu moins efficace que le stockage des données non compressées. Une logique similaire s'applique à la version de 16 bits.

Si la différence entre deux valeurs dépasse la plage de 1 octet (DELTA) ou de 2 octets (DELTA32K), la valeur d'origine complète est stockée, avec un indicateur de 1 octet en tête. La plage de 1 octet s'étend de -127 à 127, et la plage de 2 octets, de -32K à 32K.

Le tableau suivant présente le fonctionnement d'un encodage Delta pour une colonne numérique.

Valeur de données d'origine	Taille d'origine (octets)	Différence (delta)	Valeur compressée	Taille compressée (octets)
1	4		1	1 + 4 (indicateur + valeur réelle)
5	4	4	4	1
50	4	45	45	1
200	4	150	150	1 + 4 (indicateur + valeur réelle)
185	4	-15	-15	1
220	4	35	35	1
221	4	1	1	1
Totaux	28			15

LZO

L'encodage LZO fournit un taux de compression très élevé avec de bonnes performances. L'encodage LZO fonctionne particulièrement bien pour les colonnes CHAR et VARCHAR qui stockent de très longues chaînes de caractères. Ils sont particulièrement adaptés au texte libre, comme les descriptions de produits, les commentaires des utilisateurs ou les chaînes JSON.

Mostly

Les encodages Mostly sont utiles lorsque le type de données d'une colonne est plus volumineux que la plupart des valeurs stockées le nécessitent. En spécifiant un encodage Mostly pour ce type de colonne, vous pouvez compresser la majorité des valeurs de la colonne dans une taille

de stockage standard inférieure. Les autres valeurs qui ne peuvent pas être compressées sont stockées dans leur format brut. Par exemple, vous pouvez compresser une colonne 16 bits, telle qu'une INT2 colonne, en stockage 8 bits.

En général, les encodages Mostly fonctionnent avec les types de données suivants :

- SMALLINT/ INT2 (16 bits)
- INTEGER/INT (32 bits)
- BIGINT/ INT8 (64 bits)
- DECIMAL/NUMERIC (64 bits)

Choisissez la variation appropriée de l’encodage Mostly en fonction de la taille du type de données de la colonne. Par exemple, appliquez MOSTLY8 à une colonne définie comme une colonne entière de 16 bits. L'application MOSTLY16 à une colonne avec un type de données 16 bits ou MOSTLY32 à une colonne avec un type de données 32 bits n'est pas autorisée.

Les encodages Mostly peuvent être moins efficaces qu’aucune compression du tout, si bon nombre des valeurs de la colonne ne peuvent pas être compressées. Avant d’appliquer l’un de ces encodages à une colonne, effectuez une vérification. La plupart des valeurs que vous allez charger maintenant (et que vous êtes susceptible de charger à l’avenir) devraient se situer dans les plages indiquées dans le tableau suivant.

Encodage	Taille de stockage compressée	Plage de valeurs qui peuvent être compressées (les valeurs situées en dehors de la plage sont stockées dans un format brut)
MOSTLY8	1 octet (8 bits)	-128 à 127
MOSTLY16	2 octets (16 bits)	-32768 à 32767
MOSTLY32	4 octets (32 bits)	-2147483648 à +2147483647

> **Note**
>
> Pour les valeurs décimales, ignorez la virgule pour déterminer si la valeur correspond à la plage. Par exemple, 1 234,56 est traité comme 123 456 et peut être compressé dans une colonne. MOSTLY32

Par exemple, la colonne VENUEID de la table VENUE est définie comme une colonne de nombres entiers bruts, ce qui signifie que ses valeurs consomment 4 octets de stockage. Toutefois, la plage de valeurs actuelle de la colonne s'étend de **`0`** à **`309`**. Par conséquent, la recréation et le rechargement de cette table avec le MOSTLY16 codage pour VENUEID réduiraient le stockage de chaque valeur de cette colonne à 2 octets.

Si les valeurs VENUEID référencées dans une autre table étaient généralement comprises entre 0 et 127, il peut être judicieux de coder cette colonne de clé étrangère en tant que. MOSTLY8 Avant de faire votre choix, exécutez plusieurs requêtes sur les données de la table de référencement pour savoir si les valeurs se situent principalement dans une plage de 8, 16 ou 32 bits

Le tableau suivant indique les tailles compressées pour des valeurs numériques spécifiques lorsque les MOSTLY32 codages MOSTLY8 MOSTLY16, et sont utilisés :

Valeur d'origine	Taille INT ou BIGINT d'origine (octets)	MOSTLY8 taille compressée (octets)	MOSTLY16 taille compressée (octets)	MOSTLY32 taille compressée (octets)
1	4	1	2	4
10	4	1	2	4
100	4	1	2	4
1 000	4	Identique à la taille des données brutes	2	4
10 000	4		2	4
20 000	4		2	4
40 000	8		Identique à la taille des données brutes	4

Valeur d'origine	Taille INT ou BIGINT d'origine (octets)	MOSTLY8 taille compressée (octets)	MOSTLY16 taille compressée (octets)	MOSTLY32 taille compressée (octets)
100 000	8			4
2 000 000 000	8			4

Run length

L'encodage de la longueur d'exécution remplace une valeur qui est répétée de façon consécutive par un jeton composé de la valeur et d'un calcul du nombre d'occurrences consécutives (la longueur de l'exécution). Un dictionnaire distinct de valeurs uniques est créé pour chaque bloc de valeurs de la colonne sur le disque. (Un bloc de disque Amazon Redshift occupe 1 Mo.) Cet encodage est mieux adapté à une table dans laquelle les valeurs de données sont souvent répétées de façon consécutive, par exemple, lorsque la table est triée en fonction de ces valeurs.

Supposons par exemple qu'une colonne d'une grande table de dimension présente un petit domaine prévisible, comme une colonne COLOR avec moins de 10 valeurs possibles. Ces valeurs sont susceptibles de tomber en longues séquences dans le tableau, même si les données ne sont pas triées.

Nous ne recommandons pas d'appliquer l'encodage de la longueur d'exécution sur une colonne désignée comme clé de tri. Les analyses restreintes à la plage sont plus performantes lorsque les blocs contiennent un nombre de lignes similaire. Si les colonnes de clés sont beaucoup plus compressées que les autres colonnes de la même requête, des analyses à plage restreintes peuvent avoir de mauvaises performances.

Le tableau suivant utilise l'exemple de la colonne COLOR pour montrer comment fonctionne le codage de la longueur d'exécution.

Valeur de données d'origine	Taille d'origine (octets)	Valeur compressée (jeton)	Taille compressée (octets)
Blue	4	{2,bleu}	5
Blue	4		0

Valeur de données d'origine	Taille d'origine (octets)	Valeur compressée (jeton)	Taille compressée (octets)
Green	5	{3,vert}	6
Green	5		0
Green	5		0
Blue	4	{1,Blue}	5
Yellow	6	{4,Yellow}	7
Yellow	6		0
Yellow	6		0
Yellow	6		0
Total	51		23

Text255 and Text32k

Les encodages text255 et text32k sont utiles pour la compression de colonnes VARCHAR contenant des mots récurrents. Un dictionnaire distinct de mots uniques est créé pour chaque bloc de valeurs de la colonne sur le disque. (Un bloc de disque Amazon Redshift occupe 1 Mo.) Le dictionnaire contient les 245 premiers mots uniques de la colonne. Ces termes sont remplacés sur le disque par une valeur d'index d'un octet représentant l'une des 245 valeurs, et tous les mots qui ne sont pas représentées dans le dictionnaire sont stockés sous une forme décompressée. Le processus se répète pour chaque bloc de disque de 1 Mo. Si les mots indexés apparaissent fréquemment dans la colonne, celle-ci présente un taux de compression élevé.

Pour l'encodage text32k, le principe est le même, mais le dictionnaire de chaque bloc ne capture pas un nombre de mots spécifique. A la place, le dictionnaire indexe chaque mot unique qu'il trouve jusqu'à ce que les entrées combinées atteignent une longueur de 32K, moins le dépassement. Les valeurs d'index sont stockées dans deux octets.

Par exemple, prenons la colonne VENUENAME de la table VENUE. Les mots comme **`Arena`**, **`Center`** et **`Theatre`** sont récurrents dans cette colonne et sont susceptibles de figurer parmi les 245 premiers mots rencontrés dans chaque bloc si la compression text255 est appliquée.

Si c’est le cas, cette colonne bénéficie d’une compression. En effet, chaque fois que ces mots apparaissent, ils n’occupent qu’un seul octet de stockage (au lieu de 5, 6 ou 7 octets, respectivement).

ZSTD

L’encodage Zstandard (ZSTD) fournit un taux de compression élevé avec de très bonnes performances pour des ensembles de données variés. ZSTD fonctionne particulièrement bien avec les colonnes CHAR et VARCHAR qui stockent à large éventail de chaînes longues et courtes, telles que des descriptions de produits, des commentaires d’utilisateurs, des journaux ou des chaînes JSON. Lorsque certains algorithmes, tels que le codage Delta ou le codage principalement, peuvent potentiellement utiliser plus d'espace de stockage que l'absence de compression, il est peu probable que le ZSTD augmente l'utilisation du disque.

ZSTD prend en charge les types de données SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP et TIMESTAMPTZ.

Le tableau suivant identifie le codage de compression pris en charge et les types de données qui prennent en charge l’encodage.

Type d’encodage	Mot clé dans CREATE TABLE et ALTER TABLE	Types de données
Brut (aucune compression)	RAW	Tous
AZ64	AZ64	SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIMESTAMP, TIMESTAMPTZ
Dictionnaire d’octets	BYTEDICT	SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMP TZ
Delta	DELTA DELTA32 K	SMALLINT, INT, BIGINT, DATE, TIMESTAMP, DECIMAL

Type d’encodage	Mot clé dans CREATE TABLE et ALTER TABLE	Types de données
		INT, BIGINT, DATE, TIMESTAMP, DECIMAL
LZO	LZO	SMALLINT, INTEGER, BIGINT, DECIMAL, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMP TZ, SUPER
Mostlyn	MOSTLY8	SMALLINT, INT, BIGINT, DECIMAL
	MOSTLY16	INT, BIGINT, DECIMAL
	MOSTLY32	BIGINT, DECIMAL
Run-length	RUNLENGTH	SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMPTZ
Texte	TEXT255	VARCHAR uniquement
	TEXT32 K	VARCHAR uniquement
Zstandard	ZSTD	SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP, TIMESTAMPTZ, SUPER

Test des encodages de compression

Si vous souhaitez spécifier manuellement les encodages de colonnes, vous pouvez tester différents encodages avec vos données.

> **Note**
>
> Nous recommandons d'utiliser la commande COPY pour charger les données autant que possible et de permettre à la commande COPY de choisir les encodages optimaux en fonction de vos données. Vous pouvez également utiliser la commande ANALYZE COMPRESSION pour afficher les encodages suggérés pour les données existantes. Pour plus de détails sur la mise en œuvre de la compression automatique, consultez Chargement des tables avec compression automatique.

Pour effectuer un test pertinent de compression des données, vous devez disposer d'un grand nombre de lignes. Dans cet exemple, nous créons une table et insérons des lignes en utilisant une instruction qui effectue une sélection à partir de deux tables : VENUE et LISTING. Nous laissons de côté la clause WHERE qui devrait normalement joindre les deux tables. Le résultat est que chaque ligne de la table VENUE est jointe à toutes les lignes de la table LISTING, soit un total de plus de 32 millions de lignes. Cette méthode est connue sous le nom de jointure cartésienne et n'est normalement pas recommandée. Toutefois, dans le cas présent, il s'agit d'une méthode pratique pour créer de nombreuses lignes. Si vous disposez d'une table existante contenant des données que vous souhaitez tester, vous pouvez ignorer cette étape.

Après avoir obtenu une table avec des données types, nous créons une table avec sept colonnes. Chacune a un encodage de compression différent : raw, bytedict, lzo, run length, text255, text32k et zstd. Nous remplissons chaque colonne avec exactement les mêmes données en exécutant une commande INSERT qui sélectionne les données de la première table.

Pour tester les encodages de compression, procédez comme suit :

1. (Facultatif) Tout d'abord, utilisez une jointure cartésienne pour créer une table avec un grand nombre de lignes. Ignorez cette étape si vous voulez tester une table existante.

```
create table cartesian_venue(
venueid smallint not null distkey sortkey,
venuename varchar(100),
venuecity varchar(30),
venuestate char(2),
venueseats integer);

insert into cartesian_venue
select venueid, venuename, venuecity, venuestate, venueseats
```

```
from venue, listing;
```

2. Ensuite, nous créons une table avec les encodages que vous voulez comparer.

```
create table encodingvenue (
venueraw varchar(100) encode raw,
venuebytedict varchar(100) encode bytedict,
venuelzo varchar(100) encode lzo,
venuerunlength varchar(100) encode runlength,
venuetext255 varchar(100) encode text255,
venuetext32k varchar(100) encode text32k,
venuezstd varchar(100) encode zstd);
```

3. Insérez les mêmes données dans toutes les colonnes à l'aide d'une instruction INSERT avec une clause SELECT.

```
insert into encodingvenue
select venuename as venueraw, venuename as venuebytedict, venuename as venuelzo,
 venuename as venuerunlength, venuename as  venuetext32k, venuename as  venuetext255,
 venuename as venuezstd
from cartesian_venue;
```

4. Vérifiez le nombre de lignes de la nouvelle table.

```
select count(*) from encodingvenue

  count
----------
 38884394
(1 row)
```

5. Interrogez la table système STV_BLOCKLIST pour comparer le nombre de blocs de disque de 1 Mo utilisés par chaque colonne.

   La fonction d'agrégation MAX renvoie le plus grand nombre de blocs pour chaque colonne. La table STV_BLOCKLIST inclut les détails de trois colonnes générées par le système. Cet exemple utilise `col < 6` dans la clause WHERE pour exclure les colonnes générées par le système.

```
select col, max(blocknum)
from stv_blocklist b, stv_tbl_perm p
where (b.tbl=p.id) and name ='encodingvenue'
and col < 7
```

```
group by name, col
order by col;
```

La requête renvoie les résultats suivants. Les colonnes sont numérotés en commençant par zéro. En fonction de la configuration de votre cluster, votre résultat peut comporter des nombres différents, mais les tailles relatives doivent être similaires. Vous pouvez voir que l'encodage BYTEDICT sur la deuxième colonne a produit les meilleurs résultats pour cet ensemble de données. Cette approche a un taux de compression supérieur à 20:1. Les encodages LZO et ZSTD produisent également d'excellents résultats. Des jeux de données différents produisent des résultats différents, bien sûr. LZO produit souvent les meilleurs résultats de compression, lorsqu'une colonne contient des chaînes de texte plus longues.

```
 col | max
-----+-----
   0 | 203
   1 |  10
   2 |  22
   3 | 204
   4 |  56
   5 |  72
   6 |  20
(7 rows)
```

Si vous disposez de données dans une table existante, vous pouvez utiliser la commande ANALYZE COMPRESSION pour afficher l'encodage suggéré pour la table. Par exemple, l'exemple suivant illustre l'encodage recommandé pour une copie de la table VENUE, CARTESIAN_VENUE, qui contient 38 millions de lignes. Notez que la commande ANALYZE COMPRESSION recommande l'encodage LZO pour la colonne VENUENAME. ANALYZE COMPRESSION choisit la compression optimale en fonction de plusieurs facteurs, notamment la réduction du pourcentage. Dans ce cas spécifique, BYTEDICT offre une meilleure compression, mais LZO produit également une compression supérieure à 90 %.

```
analyze compression cartesian_venue;

Table          | Column     | Encoding | Est_reduction_pct
---------------+------------+----------+------------------
reallybigvenue | venueid    | lzo      | 97.54
reallybigvenue | venuename  | lzo      | 91.71
reallybigvenue | venuecity  | lzo      | 96.01
```

```
reallybigvenue | venuestate | lzo      | 97.68
reallybigvenue | venueseats | lzo      | 98.21
```

exemple

L'exemple suivant crée une table CUSTOMER comportant des colonnes contenant différents types de données. Cette instruction CREATE TABLE présente l'une des nombreuses combinaisons d'encodages de compression possibles pour ces colonnes.

```
create table customer(
custkey int encode delta,
custname varchar(30) encode raw,
gender varchar(7) encode text255,
address varchar(200) encode text255,
city varchar(30) encode text255,
state char(2) encode raw,
zipcode char(5) encode bytedict,
start_date date encode delta32k);
```

Le tableau suivant présente les encodages de colonne qui ont été choisi pour la table CUSTOMER et fournit une explication pour chaque choix :

Colonne	Type de données	Encodage	Explication
CUSTKEY	int	delta	CUSTKEY consiste en valeurs de nombres entiers consécutifs uniques. Du fait que les différences seront d'un octet, DELTA est un bon choix.
CUSTNAME	varchar(30)	raw	CUSTNAME dispose d'un grand domaine avec peu de valeurs répétées. N'importe quel encodage de compression

Colonne	Type de données	Encodage	Explication
			serait probablement inefficace.
GENDER	varchar(7)	text255	GENDER est un très petit domaine avec de nombreuses valeurs répétées. Text255 fonctionne également bien avec les colonnes VARCHAR dans laquelle les mêmes mots se répètent.
ADDRESS	varchar(200)	text255	ADDRESS est un grand domaine, mais il contient beaucoup de mots répétés, comme Rue, Avenue, Nord, Sud, etc. Text 255 et text 32k sont utiles pour la compression de colonnes VARCHAR dans lesquelles les mêmes mots se répètent. La longueur de la colonne est courte, text255 est donc un bon choix.

Colonne	Type de données	Encodage	Explication
CITY	varchar(30)	text255	CITY est un grand domaine, avec certaines valeurs répétées. Certains noms de villes sont utilisés beaucoup plus fréquemment que d'autres. Text255 est un bon choix pour les mêmes raisons que ADDRESS.
STATE	char(2)	raw	Aux États-Unis, STATE est un domaine précis composé de 50 valeurs de deux caractères. L'encodage Bytedict entraînerait une compression, mais du fait que la taille de la colonne n'est que deux caractères, la compression ne vaut peut-être pas la surcharge que représente la décompression des données.

Colonne	Type de données	Encodage	Explication
ZIPCODE	char(5)	bytedict	ZIPCODE est un domaine connu composés de moins de 50 000 valeurs uniques. Certains codes postaux se répètent beaucoup plus souvent que d'autres. L'encodag e bytedict est très efficace lorsqu'un e colonne contient un nombre limité de valeurs uniques.
START_DATE	date	delta32k	Les encodages delta sont très utiles pour les colonnes de date et d'heure, surtout si les lignes sont chargées dans l'ordre des dates.

Distribution des données pour l'optimisation des requêtes

Lorsque vous chargez des données dans une table, Amazon Redshift distribue les lignes de la table à chacun des nœuds de calcul en fonction du style de distribution de la table. Lorsque vous exécutez une requête, l'optimiseur de requête redistribue les lignes sur les nœuds de calcul en fonction des besoins afin d'effectuer les jointures et les agrégations. Le choix d'un style de distribution de table a pour objectif de minimiser l'impact de l'étape de redistribution en plaçant les données où elles doivent être avant que l'exécution de la requête.

> **Note**
>
> Cette section vous présente les principes de distribution des données dans une base de données Amazon Redshift. Nous vous recommandons de créer vos tables avec `DISTSTYLE AUTO`. Si vous procédez ainsi, Amazon Redshift utilise l'optimisation automatique des tables pour choisir le style de distribution des données. Pour plus d'informations, consultez Optimisation automatique des tables. Le reste de cette section fournit des détails sur les styles de distribution.

Rubriques

- Concepts de distribution de données
- Styles de distribution
- Affichage des styles de distribution
- Évaluation des modèles de requêtes
- Détermination des styles de distribution
- Évaluation du plan de requête
- Exemple de plan de requête
- Exemples de distribution

Concepts de distribution de données

Voici quelques concepts de distribution de données pour Amazon Redshift.

Nœuds et tranches

Un cluster Amazon Redshift est un ensemble de nœuds. Chaque nœud du cluster a ses propres système d'exploitation, mémoire dédiée et stockage sur disque dédié. Un des nœud est le nœud principal, qui gère la distribution des données et des tâches de traitement des requêtes dans les nœuds de calcul. Les nœuds de calcul fournissent les ressources nécessaires à l'exécution de ces tâches.

Le stockage sur disque d'un nœud de calcul est divisé en un certain nombre de tranches. Le nombre de tranches par nœud dépend de la taille de nœud du cluster. Les nœuds participent tous à l'exécution de requêtes parallèles, en travaillant sur des données réparties aussi uniformément que possible sur les tranches. Pour plus d'informations sur le nombre de tranches pour chaque taille de

nœud, consultez la rubrique À propos des clusters et nœuds dans le Guide de la gestion du cluster Amazon Redshift.

Redistribution des données

Lorsque vous chargez des données dans une table, Amazon Redshift distribue les lignes de la table sur chacune des tranches de nœuds en fonction du style de distribution de la table. Dans le cadre d'un plan de requête, l'optimiseur détermine où les blocs de données doivent être situés pour exécuter au mieux la requête. Les données sont ensuite déplacées physiquement, ou redistribuées, pendant l'exécution de la requête. La redistribution peut impliquer l'envoi de lignes spécifiques aux nœuds pour joindre ou diffuser une table entière sur tous les nœuds.

La redistribution de données peut représenter une partie importante du coût d'un plan de requête, et le trafic réseau qu'elle génère peut affecter d'autres opérations de base de données et ralentir les performances globales du système. Tant que vous prévoyez le meilleur emplacement pour situer les données à l'origine, vous pouvez réduire l'impact de la redistribution des données.

Objectifs de la distribution de données

Lorsque vous chargez des données dans une table, Amazon Redshift distribue les lignes de la table sur les nœuds de calcul et les tranches en fonction du style de distribution que vous avez choisi lorsque vous avez créé la table. La distribution de données a deux objectifs principaux :

- Distribuer la charge de travail de manière uniforme entre les nœuds du cluster. La distribution inégale, ou l'asymétrie de la distribution de données, oblige certains nœuds à travailler plus que d'autres, ce qui nuit aux performances des requêtes.
- Minimiser les déplacements de données lors de l'exécution d'une requête. Si les lignes qui participent aux jointures ou aux agrégats sont déjà colocalisées sur les nœuds avec leurs lignes de jonction dans d'autres tables, l'optimiseur n'a pas besoin de redistribuer autant de données lors de l'exécution des requêtes.

La stratégie de distribution que vous choisissez pour votre base de données a des conséquences importantes sur les performance des requêtes, les besoins en stockage, le chargement des données et la maintenance. En choisissant le meilleur style de distribution pour chaque table, vous pouvez équilibrer la distribution de vos données et améliorer considérablement les performances globales du système.

Styles de distribution

Lorsque vous créez une table, vous pouvez désigner l'un des styles de distribution suivants : AUTO, EVEN, KEY ou ALL.

Si vous ne spécifiez pas de style de distribution, Amazon Redshift utilise la distribution AUTO.

Distribution AUTO

Avec la distribution AUTO, Amazon Redshift attribue un style de distribution optimal en fonction de la taille des données de la table. Par exemple, si le style de distribution AUTO est spécifié, Amazon Redshift affecte initialement le style de distribution ALL à une petite table. Lorsque la table s'agrandit, Amazon Redshift peut modifier le style de distribution sur KEY et choisir la clé primaire (ou une colonne de la clé primaire composite) comme clé de distribution. Si la table s'agrandit et qu'aucune des colonnes ne peut être la clé de distribution, Amazon Redshift change le style de distribution sur EVEN. La modification de style de distribution se produit en arrière-plan et son impact sur les requêtes des utilisateurs est minimal.

Pour afficher les actions qu'Amazon Redshift a effectuées automatiquement pour modifier une clé de distribution de table, consultez SVL_AUTO_WORKER_ACTION. Pour afficher les recommandations actuelles concernant la modification de la clé de distribution d'une table, consultez SVV_ALTER_TABLE_RECOMMENDATIONS.

Pour afficher le style de distribution appliqué à une table, interrogez la vue du catalogue système PG_CLASS_INFO. Pour plus d'informations, consultez Affichage des styles de distribution. Si vous ne spécifiez pas de style de distribution avec l'instruction CREATE TABLE, Amazon Redshift applique la distribution AUTO.

Distribution EVEN

Le nœud principal distribue les lignes entre les tranches selon le principe du tourniquet (round robin), quelles que soient les valeurs d'une colonne donnée. La distribution Even est appropriée lorsqu'une table ne participe pas aux jointures. Elle est également appropriée lorsqu'il n'y a pas de choix clair entre une distribution KEY et une distribution ALL.

Distribution KEY

Les lignes sont distribuées en fonction des valeurs dans une colonne. Le nœud principal place des valeurs correspondantes sur la même tranche de nœud. Si vous distribuez une paire de tables sur

les clés de jonction, le nœud principal colocalise les lignes sur les tranches en fonction des valeurs des colonnes de jonction. De cette façon, les valeurs correspondantes des colonnes communes sont physiquement stockées ensemble.

Distribution ALL

Une copie de la table complète est distribuée à chaque nœud. Là où la distribution EVEN ou la distribution KEY place uniquement une partie des lignes de la table sur chaque nœud, la distribution ALL vérifie que chaque ligne est colocalisée pour chaque jointure à laquelle la table participe.

La distribution ALL multiplie le stockage requis par le nombre de nœuds du cluster. Il faut donc beaucoup plus de temps pour charger, mettre à jour ou insérer des données dans plusieurs tables. La distribution ALL convient uniquement aux tables relativement lentes ; c'est-à-dire les tables qui ne sont pas mises à jour fréquemment ou de façon extensive. Le coût de la redistribution de petites tables au cours d'une requête étant faible, il n'y a pas d'avantage significatif à définir des petites tables de dimension comme DISTSTYLE ALL.

> **Note**
>
> Une fois que vous avez spécifié un style de distribution pour une colonne, Amazon Redshift gère la distribution des données au niveau du cluster. Amazon Redshift ne requiert ni ne prend en charge le concept de partitionnement des données au sein des objets de la base de données. Vous n'avez pas besoin de créer des tablespaces ou de définir des schémas de partitionnement pour les tables.

Dans certains scénarios, vous pouvez modifier le style de distribution d'une table une fois qu'elle a été créée. Pour plus d'informations, consultez ALTER TABLE. Pour les scénarios dans lesquels vous ne pouvez pas modifier le style de distribution d'une table après sa création, vous pouvez recréer la table et la remplir avec une copie complète. Pour plus d'informations, consultez Exécution d'une copie complète

Affichage des styles de distribution

Pour afficher le style de distribution d'une table, interrogez la vue PG_CLASS_INFO ou SVV_TABLE_INFO.

La colonne RELEFFECTIVEDISTSTYLE de PG_CLASS_INFO indique le style de distribution actuel pour la table. Si la table utilise la distribution automatique, RELEFFECTIVEDISTSTYLE a pour valeur

10, 11 ou 12, ce qui indique si le style de distribution effectif est AUTO (ALL), AUTO (EVEN) ou AUTO (KEY). Si la table utilise la distribution automatique, le style de distribution peut initialement afficher AUTO (ALL), puis passer à AUTO (EVEN) ou AUTO (KEY) lorsque la table se développe.

La table suivante donne le style de distribution pour chaque valeur de la colonne RELEFFECTIVEDISTSTYLE :

RELEFFECTIVEDISTSTYLE	Style de distribution actuel
0	EVEN
1	KEY
8	ALL
10	AUTO (ALL)
11	AUTO (EVEN)
12	AUTO (KEY)

La colonne DISTSTYLE dans SVV_TABLE_INFO indique le style de distribution actuel de la table. Si la table utilise la distribution automatique, DISTSTYLE a pour valeur AUTO (ALL), AUTO (EVEN) ou AUTO (KEY).

L'exemple suivant crée quatre tables en utilisant les trois styles de distribution et la distribution automatique, puis interroge SVV_TABLE_INFO pour afficher les styles de distribution.

```
create table public.dist_key (col1 int)
diststyle key distkey (col1);

insert into public.dist_key values (1);

create table public.dist_even (col1 int)
diststyle even;

insert into public.dist_even values (1);

create table public.dist_all (col1 int)
diststyle all;
```

```
insert into public.dist_all values (1);

create table public.dist_auto (col1 int);

insert into public.dist_auto values (1);

select "schema", "table", diststyle from SVV_TABLE_INFO
where "table" like 'dist%';

        schema   |    table        | diststyle
     ------------+-----------------+------------
      public     | dist_key        | KEY(col1)
      public     | dist_even       | EVEN
      public     | dist_all        | ALL
      public     | dist_auto       | AUTO(ALL)
```

Évaluation des modèles de requêtes

Le choix des styles de distribution n'est qu'un des aspects de la conception de base de données. Prenez les styles de distribution dans le contexte du système entier, en équilibrant la distribution avec d'autres facteurs importants tels que la taille du cluster, les méthodes d'encodage de compression, les clés de tri et les contraintes de table.

Testez votre système avec des données qui sont aussi proches que possible des données réelles.

Pour faire de bons choix en matière de styles de distribution, vous devez comprendre les modèles de requête de votre application Amazon Redshift. Identifiez les requêtes les plus coûteuses de votre système et fondez votre conception de base de données initiale sur les exigences de ces requêtes. Les facteurs qui déterminent le coût total d'une requête comprennent la durée d'exécution de la requête et les ressources informatiques qu'elle consomme. Les autres facteurs qui déterminent le coût d'une requête sont la fréquence d'exécution et la perturbation des autres requêtes et des opérations de la base de données.

Identifiez les tables utilisées par les requêtes les plus coûteuses et évaluez leur rôle dans la durée d'exécution des requêtes. Réfléchissez à la manière dont les tables sont jointes et regroupées.

Utilisez les instructions fournies dans cette section pour choisir un style de distribution pour chaque table. Une fois cela fait, créez les tables et chargez-les avec des données aussi proches que possible des données réelles. Testez ensuite les tables pour les types de requêtes que vous pensez utiliser. Vous pouvez évaluer les plans EXPLAIN de la requête pour identifier des opportunités de réglage.

Comparez les temps de chargement, l'espace de stockage et les durées d'exécution des requêtes pour équilibrer les configurations requises de votre système.

Détermination des styles de distribution

Les considérations et recommandations relatives à la désignation des styles distribution dans cette section sont basées sur un schéma en étoile à titre d'exemple. La conception de votre base de données peut être basée sur un schéma en étoile, une variante de ce schéma ou un schéma totalement différent. Amazon Redshift est conçu pour fonctionner efficacement avec n'importe quel conception de schéma, à votre choix. Les principes de cette section peuvent être appliqués à n'importe quel schéma de conception.

1. Spécifiez la clé primaire et les clés étrangères de toutes vos tables.

   Amazon Redshift n'applique pas les contraintes de clés primaires et de clés étrangères, mais l'optimiseur de requêtes les utilise lorsqu'il génère des plans de requêtes. Si vous définissez les clés primaires et les clés étrangères, votre application doit gérer la validité des clés.
2. Distribuez la table des faits et sa plus grande table de dimension sur leurs colonnes communes.

   Choisissez la dimension la plus grande en fonction de la taille du jeu de données qui participe à la jointure la plus courante, et pas seulement de la taille de la table. Si une table est généralement filtrée, à l'aide d'une clause WHERE, seule une partie de ses lignes participe à la jointure. Une telle table a moins d'impact sur la redistribution qu'une table plus petite qui comporte davantage de données. Désignez la clé primaire de la table de dimension et la clé étrangère correspondante de la table des faits en tant que DISTKEY. Si plusieurs tables utilisent la même clé de distribution, elles seront également colocalisées avec la table des faits. Votre table de faits peut n'avoir qu'une seule clé de distribution. Toutes les tables qui sont jointes à une autre clé de distribution ne sont pas colocalisées avec la table des faits.
3. Désignez des clés de distribution pour les autres tables de dimension.

   Distribuez les tables sur leurs clés primaires ou leurs clés étrangères, en fonction de la manière dont elles sont jointes le plus souvent avec d'autres tables.
4. Évaluez s'il faut modifier certaines des tables de dimension pour utiliser la distribution ALL.

   Si une table de dimension ne peut pas être colocalisée avec la table des faits ou d'autres tables de jointure importantes, il vous est possible d'améliorer considérablement les performances des requêtes en distribuant la totalité de la table sur tous les nœuds. La distribution ALL multiplie les besoins en espace de stockage et augmente les temps de chargement et les opérations de

maintenance. Vous devez donc évaluer tous les facteurs avant de choisir la distribution ALL. La section suivante explique comment identifier les candidats à la distribution ALL en évaluant le plan EXPLAIN.

5. Utilisez la distribution AUTO pour les tables restantes.

   Si une table est largement dénormalisée et ne participe pas à des jointures, ou si vous n'avez pas fait de choix arrêté sur un autre style de distribution, utilisez la distribution AUTO.

Pour laisser Amazon Redshift choisir le style de distribution approprié, ne spécifiez pas explicitement de style de distribution.

Évaluation du plan de requête

Vous pouvez utiliser les plans de requête pour identifier des candidats à l'optimisation du style de distribution.

Après avoir pris vos décisions de conception initiales, créez vos tables, chargez-les avec des données et testez-les. Utilisez un ensemble de données de test aussi proche que possible des données réelles. Mesurez les temps de chargement à utiliser comme référence pour les comparaisons.

Évaluez les requêtes qui sont représentatives des requêtes les plus coûteuses que vous vous attendez à exécuter, en particulier les requêtes qui utilisent des jointures et des agrégations. Comparez les durées d'exécution pour différentes options de conception. Lorsque vous comparez les durées d'exécution, ne comptez pas la première fois que la requête est exécutée, car la première exécution inclut le temps de compilation.

DS_DIST_NONE

Aucun redistribution n'est obligatoire, car les tranches correspondantes sont situées sur les nœuds de calcul. Vous n'avez généralement qu'une seule étape DS_DIST_NONE, la jointure entre la table de faits et une table de dimension.

DS_DIST_ALL_NONE

Aucun redistribution n'est obligatoire, car la table de jointure interne a utilisé DISTSTYLE ALL. La totalité de la table se trouve sur chaque nœud.

DS_DIST_INNER

La table interne est redistribuée.

DS_DIST_OUTER

La table externe est redistribuée.

DS_BCAST_INNER

Une copie de la totalité de la table interne est diffusée à tous les nœuds de calcul.

DS_DIST_ALL_INNER

La totalité de la table interne est redistribué à une seule tranche, car la table externe utilise DISTSTYLE ALL.

DS_DIST_BOTH

Les deux tables sont redistribuées.

DS_DIST_NONE et DS_DIST_ALL_NONE sont corrects. Ils indiquent qu'aucune distribution n'a été requise pour cette étape, car toutes les jointures sont colocalisées.

DS_DIST_INNER signifie que l'étape aura probablement un coût relativement élevé, car la table interne est redistribuée aux nœuds. DS_DIST_INNER indique que la table externe est déjà correctement distribuée sur la clé de jointure. Définissez la clé de distribution de la table interne sur la clé de jointure pour convertir celle-ci en DS_DIST_NONE. Dans certains cas, la distribution de la table interne sur la clé de jointure n'est pas possible parce que la table externe n'est pas distribuée sur la clé de jointure. Si c'est le cas, évaluez s'il faut utiliser la distribution ALL pour la table interne. Si la table n'est pas mise à jour fréquemment ou de manière extensive, et qu'elle est suffisamment grande pour supporter un coût de redistribution élevé, changez le style de distribution à ALL et testez à nouveau. La distribution ALL entraîne des temps de charge supérieurs. Par conséquent, lorsque vous refaites le test, incluez le temps de chargement dans vos facteurs d'évaluation.

DS_DIST_ALL_INNER n'est pas correct. Cela signifie que la totalité de la table interne est redistribué à une seule tranche, car la table externe utilise DISTSTYLE ALL, de sorte qu'une copie de la totalité de la table externe est située sur chaque nœud. Il en découle une exécution en série inefficace de la jointure sur un seul nœud, au lieu de tirer parti de l'exécution en parallèle sur l'ensemble des nœuds. DISTSTYLE ALL est conçu pour être utilisé uniquement pour la table de jointure interne. Au lieu de cela, spécifiez une clé de distribution ou utilisez la distribution uniforme pour la table externe.

DS_BCAST_INNER et DS_DIST_BOTH ne sont pas corrects. En général, ces redistributions surviennent parce que les tables ne sont pas jointes sur leurs clés de distribution. Si la table des faits ne dispose pas déjà d'une clé de distribution, définissez la colonne de jointure comme clé

de distribution pour les deux tables. Si la table de faits a déjà une clé de distribution sur une autre colonne, évaluez si le fait de changer la clé de distribution pour colocaliser cette jointure améliore les performances globales. Si la modification de la clé de distribution de la table externe n'est pas un choix optimal, vous pouvez obtenir la collocation en spécifiant DISTSTYLE ALL pour la table interne.

L'exemple suivant illustre une partie d'un plan de requête avec les étiquettes DS_BCAST_INNER et DS_DIST_NONE.

```
->  XN Hash Join DS_BCAST_INNER  (cost=112.50..3272334142.59 rows=170771 width=84)
        Hash Cond: ("outer".venueid = "inner".venueid)
        ->  XN Hash Join DS_BCAST_INNER  (cost=109.98..3167290276.71 rows=172456
 width=47)
              Hash Cond: ("outer".eventid = "inner".eventid)
              ->  XN Merge Join DS_DIST_NONE  (cost=0.00..6286.47 rows=172456 width=30)
                    Merge Cond: ("outer".listid = "inner".listid)
                    ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497
 width=14)
                    ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=24)
```

Après avoir modifié les tables de dimension afin d'utiliser DISTSTYLE ALL, le plan de requête de la même requête affiche DS_DIST_ALL_NONE à la place de DS_BCAST_INNER. De plus, le coût relatif des étapes de la jointure ont considérablement changé. Le coût total est de `14142.59`, alors que celui de la requête précédente était de `3272334142.59`.

```
->  XN Hash Join DS_DIST_ALL_NONE  (cost=112.50..14142.59 rows=170771 width=84)
        Hash Cond: ("outer".venueid = "inner".venueid)
        ->  XN Hash Join DS_DIST_ALL_NONE  (cost=109.98..10276.71 rows=172456 width=47)
              Hash Cond: ("outer".eventid = "inner".eventid)
              ->  XN Merge Join DS_DIST_NONE  (cost=0.00..6286.47 rows=172456 width=30)
                    Merge Cond: ("outer".listid = "inner".listid)
                    ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497
 width=14)
                    ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=24)
```

Exemple de plan de requête

Cet exemple illustre la façon d'évaluer un plan de requête pour trouver des opportunités permettant d'optimiser la distribution.

Exécutez la requête suivante avec une commande EXPLAIN pour obtenir un plan de requête.

```
explain
select lastname, catname, venuename, venuecity, venuestate, eventname,
month, sum(pricepaid) as buyercost, max(totalprice) as maxtotalprice
from category join event on category.catid = event.catid
join venue on venue.venueid = event.venueid
join sales on sales.eventid = event.eventid
join listing on sales.listid = listing.listid
join date on sales.dateid = date.dateid
join users on users.userid = sales.buyerid
group by lastname, catname, venuename, venuecity, venuestate, eventname, month
having sum(pricepaid)>9999
order by catname, buyercost desc;
```

Dans la base de données TICKIT, SALES est une table de faits et LISTING constitue sa dimension la plus grande. Pour colocaliser les tables, SALES est distribuée sur LISTID, qui est la clé étrangère de LISTING, qui est distribuée sur sa clé primaire, LISTID. L'exemple suivant présente les commandes CREATE TABLE pour SALES et LISTING.

```
create table sales(
 salesid integer not null,
 listid integer not null distkey,
 sellerid integer not null,
 buyerid integer not null,
 eventid integer not null encode mostly16,
 dateid smallint not null,
 qtysold smallint not null encode mostly8,
 pricepaid decimal(8,2) encode delta32k,
 commission decimal(8,2) encode delta32k,
 saletime timestamp,
 primary key(salesid),
 foreign key(listid) references listing(listid),
 foreign key(sellerid) references users(userid),
 foreign key(buyerid) references users(userid),
 foreign key(dateid) references date(dateid))
        sortkey(listid,sellerid);

create table listing(
 listid integer not null distkey sortkey,
 sellerid integer not null,
 eventid integer not null encode mostly16,
 dateid smallint not null,
 numtickets smallint not null encode mostly8,
```

```
 priceperticket decimal(8,2) encode bytedict,
 totalprice decimal(8,2) encode mostly32,
 listtime timestamp,
 primary key(listid),
 foreign key(sellerid) references users(userid),
 foreign key(eventid) references event(eventid),
 foreign key(dateid) references date(dateid));
```

Dans le plan de requête suivant, l'étape Joindre par fusion de la jointure sur SALES et LISTING affiche DS_DIST_NONE, ce qui signifie qu'aucune redistribution n'est nécessaire pour cette étape. Toutefois, en évoluant dans le plan de requête, les autres jointures internes affichent DS_BCAST_INNER, ce qui signifie que la table interne est diffusée dans le cadre de l'exécution des requêtes. Du fait qu'une seule paire de tables peut être colocalisée à l'aide de la distribution de clés, cinq tables doivent être rediffusées.

```
QUERY PLAN
XN Merge  (cost=1015345167117.54..1015345167544.46 rows=1000 width=103)
  Merge Key: category.catname, sum(sales.pricepaid)
  ->  XN Network  (cost=1015345167117.54..1015345167544.46 rows=170771 width=103)
        Send to leader
        ->  XN Sort  (cost=1015345167117.54..1015345167544.46 rows=170771 width=103)
              Sort Key: category.catname, sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=15345150568.37..15345152276.08 rows=170771
 width=103)
                    Filter: (sum(pricepaid) > 9999.00)
                     ->  XN Hash Join DS_BCAST_INNER  (cost=742.08..15345146299.10
 rows=170771 width=103)
                           Hash Cond: ("outer".catid = "inner".catid)
                           ->  XN Hash Join DS_BCAST_INNER
 (cost=741.94..15342942456.61 rows=170771 width=97)
                                 Hash Cond: ("outer".dateid = "inner".dateid)
                                 ->  XN Hash Join DS_BCAST_INNER
 (cost=737.38..15269938609.81 rows=170766 width=90)
                                       Hash Cond: ("outer".buyerid = "inner".userid)
                                       ->  XN Hash Join DS_BCAST_INNER
 (cost=112.50..3272334142.59 rows=170771 width=84)
                                             Hash Cond: ("outer".venueid =
 "inner".venueid)
                                             ->  XN Hash Join DS_BCAST_INNER
 (cost=109.98..3167290276.71 rows=172456 width=47)
                                                   Hash Cond: ("outer".eventid =
 "inner".eventid)
```

```
                                                    ->  XN Merge Join DS_DIST_NONE
(cost=0.00..6286.47 rows=172456 width=30)
                                                          Merge Cond: ("outer".listid =
"inner".listid)
                                                          ->  XN Seq Scan on listing
(cost=0.00..1924.97 rows=192497 width=14)
                                                          ->  XN Seq Scan on sales
(cost=0.00..1724.56 rows=172456 width=24)
                                                    ->  XN Hash  (cost=87.98..87.98
rows=8798 width=25)
                                                          ->  XN Seq Scan on event
(cost=0.00..87.98 rows=8798 width=25)
                                               ->  XN Hash  (cost=2.02..2.02 rows=202
width=41)
                                                    ->  XN Seq Scan on venue
(cost=0.00..2.02 rows=202 width=41)
                                          ->  XN Hash  (cost=499.90..499.90 rows=49990
width=14)
                                               ->  XN Seq Scan on users
(cost=0.00..499.90 rows=49990 width=14)
                                     ->  XN Hash  (cost=3.65..3.65 rows=365 width=11)
                                           ->  XN Seq Scan on date  (cost=0.00..3.65
rows=365 width=11)
                               ->  XN Hash  (cost=0.11..0.11 rows=11 width=10)
                                     ->  XN Seq Scan on category  (cost=0.00..0.11 rows=11
width=10)
```

Une solution consiste à modifier les tables de sorte qu'elles contiennent DISTSTYLE ALL.

```
ALTER TABLE users ALTER DISTSTYLE ALL;
ALTER TABLE venue ALTER DISTSTYLE ALL;
ALTER TABLE category ALTER DISTSTYLE ALL;
ALTER TABLE date ALTER DISTSTYLE ALL;
ALTER TABLE event ALTER DISTSTYLE ALL;
```

Réexécutez la même requête avec EXPLAIN et examinez le nouveau plan de requête. Les jointures affichent à présent DS_DIST_ALL_NONE, ce qui signifie qu'aucune redistribution n'est nécessaire, car les données ont été distribuées à chaque nœud à l'aide de DISTSTYLE ALL.

```
QUERY PLAN
XN Merge  (cost=1000000047117.54..1000000047544.46 rows=1000 width=103)
  Merge Key: category.catname, sum(sales.pricepaid)
  ->  XN Network  (cost=1000000047117.54..1000000047544.46 rows=170771 width=103)
```

```
       Send to leader
       ->  XN Sort  (cost=1000000047117.54..1000000047544.46 rows=170771 width=103)
             Sort Key: category.catname, sum(sales.pricepaid)
             ->  XN HashAggregate  (cost=30568.37..32276.08 rows=170771 width=103)
                   Filter: (sum(pricepaid) > 9999.00)
                   ->  XN Hash Join DS_DIST_ALL_NONE  (cost=742.08..26299.10
rows=170771 width=103)
                         Hash Cond: ("outer".buyerid = "inner".userid)
                         ->  XN Hash Join DS_DIST_ALL_NONE  (cost=117.20..21831.99
rows=170766 width=97)
                               Hash Cond: ("outer".dateid = "inner".dateid)
                               ->  XN Hash Join DS_DIST_ALL_NONE
(cost=112.64..17985.08 rows=170771 width=90)
                                     Hash Cond: ("outer".catid = "inner".catid)
                                     ->  XN Hash Join DS_DIST_ALL_NONE
(cost=112.50..14142.59 rows=170771 width=84)
                                           Hash Cond: ("outer".venueid =
"inner".venueid)
                                           ->  XN Hash Join DS_DIST_ALL_NONE
(cost=109.98..10276.71 rows=172456 width=47)
                                                 Hash Cond: ("outer".eventid =
"inner".eventid)
                                                 ->  XN Merge Join DS_DIST_NONE
(cost=0.00..6286.47 rows=172456 width=30)
                                                       Merge Cond: ("outer".listid =
"inner".listid)
                                                       ->  XN Seq Scan on listing
(cost=0.00..1924.97 rows=192497 width=14)
                                                       ->  XN Seq Scan on sales
(cost=0.00..1724.56 rows=172456 width=24)
                                                 ->  XN Hash  (cost=87.98..87.98
rows=8798 width=25)
                                                       ->  XN Seq Scan on event
(cost=0.00..87.98 rows=8798 width=25)
                                           ->  XN Hash  (cost=2.02..2.02 rows=202
width=41)
                                                 ->  XN Seq Scan on venue
(cost=0.00..2.02 rows=202 width=41)
                                     ->  XN Hash  (cost=0.11..0.11 rows=11 width=10)
                                           ->  XN Seq Scan on category
(cost=0.00..0.11 rows=11 width=10)
                               ->  XN Hash  (cost=3.65..3.65 rows=365 width=11)
                                     ->  XN Seq Scan on date  (cost=0.00..3.65
rows=365 width=11)
```

```
                        ->  XN Hash  (cost=499.90..499.90 rows=49990 width=14)
                              ->  XN Seq Scan on users  (cost=0.00..499.90 rows=49990
 width=14)
```

Exemples de distribution

Les exemples suivants illustrent le type de distribution des données selon les options que vous définissez dans l'instruction CREATE TABLE.

Exemples de DISTKEY

Regardez le schéma de la table USERS dans la base de données TICKIT. USERID est défini comme la colonne SORTKEY et la colonne DISTKEY :

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'users';

    column     |          type          | encoding | distkey | sortkey
---------------+------------------------+----------+---------+---------
 userid        | integer                | none     | t       |       1
 username      | character(8)           | none     | f       |       0
 firstname     | character varying(30)  | text32k  | f       |       0

...
```

USERID est un bon choix pour la colonne de distribution de cette table. Si vous interrogez la vue système SVV_DISKUSAGE, vous pouvez voir que la table est distribuée de manière très uniforme. Les nombres de colonnes sont de base zéro, USERID est donc la colonne 0.

```
select slice, col, num_values as rows, minvalue, maxvalue
from svv_diskusage
where name='users' and col=0 and rows>0
order by slice, col;

slice| col | rows  | minvalue | maxvalue
-----+-----+-------+----------+----------
0    | 0   | 12496 | 4        | 49987
1    | 0   | 12498 | 1        | 49988
2    | 0   | 12497 | 2        | 49989
3    | 0   | 12499 | 3        | 49990
(4 rows)
```

La table contient 49 990 lignes. La colonne (num_values) des lignes affiche que chaque tranche contient à peu près le même nombre de lignes. Les colonnes minvalue et maxvalue affichent la plage de valeurs sur chaque tranche. Chaque tranche inclut la quasi-totalité de la plage de valeurs. Il y a donc de fortes chances que chaque tranche participe à l'exécution d'une requête filtrant pour un groupe d'utilisateurs IDs.

Cet exemple illustre la distribution sur un petit système de test. Le nombre total de tranches est généralement beaucoup plus élevé.

Si vous utilisez habituellement la colonne STATE pour effectuer une jointure ou un regroupement, vous pouvez choisir de distribuer sur la colonne STATE. L'exemple suivant illustre un cas où vous créez une table avec les mêmes données que la table USERS mais où vous définissez la DISTKEY sur la colonne STATE. Dans ce cas, la distribution n'est pas aussi régulière. La tranche 0 (13 587 lignes) contient environ 30 % de lignes en plus que la tranche 3 (10 150 lignes). Dans une table beaucoup plus grande, ce degré d'asymétrie de la distribution peut avoir un impact négatif sur le traitement des requêtes.

```
create table userskey distkey(state) as select * from users;

select slice, col, num_values as rows, minvalue, maxvalue from svv_diskusage
where name = 'userskey' and col=0 and rows>0
order by slice, col;

slice | col | rows  | minvalue | maxvalue
------+-----+-------+----------+----------
    0 |   0 | 13587 |        5 |    49989
    1 |   0 | 11245 |        2 |    49990
    2 |   0 | 15008 |        1 |    49976
    3 |   0 | 10150 |        4 |    49986
(4 rows)
```

Exemple de DISTSTYLE EVEN

Si vous créez une table avec les mêmes données que la table USERS, mais que vous définissez DISTSTYLE sur EVEN, les lignes sont toujours distribuées uniformément entre les tranches.

```
create table userseven diststyle even as
select * from users;

select slice, col, num_values as rows, minvalue, maxvalue from svv_diskusage
where name = 'userseven' and col=0 and rows>0
```

```
order by slice, col;

slice | col | rows  | minvalue | maxvalue
------+-----+-------+----------+----------
    0 |   0 | 12497 |        4 |    49990
    1 |   0 | 12498 |        8 |    49984
    2 |   0 | 12498 |        2 |    49988
    3 |   0 | 12497 |        1 |    49989
(4 rows)
```

Cependant, du fait que la distribution n'est pas basée sur une colonne spécifique, le traitement des requêtes peut être dégradé, surtout si la table est jointe à d'autres tables. L'absence de distribution sur une colonne de jointure influence souvent le type d'opération de jointure qui peut être effectué de manière efficace. Les jointures, les agrégations et les opérations de regroupement sont optimisées lorsque les deux tables sont distribuées et triées sur leurs colonnes de jointure respectives.

Exemple de DISTSTYLE ALL

Si vous créez une table avec les mêmes données que la table USERS, mais que vous définissez DISTSTYLE sur ALL, toutes les lignes sont distribuées à la première tranche de chaque nœud.

```
select slice, col, num_values as rows, minvalue, maxvalue from svv_diskusage
where name = 'usersall' and col=0 and rows > 0
order by slice, col;

slice | col | rows  | minvalue | maxvalue
------+-----+-------+----------+----------
    0 |   0 | 49990 |        4 |    49990
    2 |   0 | 49990 |        2 |    49990

(4 rows)
```

Clés de tri

> **Note**
>
> Nous vous recommandons de créer vos tables avec `SORTKEY AUTO`. Si vous procédez ainsi, Amazon Redshift utilise l'optimisation automatique des tables pour choisir la clé de tri. Pour plus d'informations, consultez Optimisation automatique des tables. Le reste de cette section fournit des détails sur l'ordre de tri.

Lorsque vous créez une table, vous pouvez définir une ou plusieurs de ses colonnes comme clés de tri. Lorsque les données sont chargées initialement dans la table vide, les lignes sont stockées sur le disque dans un ordre trié. Les informations sur les colonnes de clés de tri sont transmises au planificateur de requête qui les utilise pour créer des plans basés sur le mode de tri des données. Pour de plus amples informations, veuillez consulter CREATE TABLE. Pour plus d'informations sur les bonnes pratiques lors de la création d'une clé de tri, consultez Choix de la meilleure clé de tri.

Le tri permet de traiter efficacement les prédicats limités à une plage. Amazon Redshift stocke les données en colonnes dans des blocs de disque de 1 Mo. Les valeurs minimale et maximale de chaque bloc sont stockées dans le cadre des métadonnées. Si une requête utilise un prédicat à plage restreinte, le processeur de requêtes peut utiliser les valeurs min et max pour ignorer rapidement un grand nombre de blocs lors des analyses de table. Supposons par exemple qu'une table stocke cinq années de données triées par date et qu'une requête spécifie une plage de dates d'un mois. Dans ce cas, vous pouvez supprimer jusqu'à 98 % des blocs du disque de l'analyse. Si les données ne sont pas triées, davantage de blocs de disque (peut-être même tous) doivent être analysés.

Vous pouvez spécifier une clé de tri composée ou une clé de tri entrelacée. Une clé de tri composée est plus efficace lorsque les prédicats de requête utilisent un préfixe, qui est un sous-ensemble des colonnes de clé de tri dans l'ordre. Une clé de tri entrelacée confère un poids égal à chaque colonne de la clé de tri, les prédicats de requête peuvent donc utiliser n'importe quel sous-ensemble des colonnes qui constituent la clé de tri, dans n'importe quel ordre.

Pour comprendre l'incidence de la clé de tri choisie sur la performance des requêtes, utilisez la commande EXPLAIN. Pour plus d'informations, consultez Workflow d'exécution et de planification de requête.

Pour définir un type de tri, utilisez le mot clé INTERLEAVED ou COMPOUND avec votre instruction CREATE TABLE ou CREATE TABLE AS. La valeur par défaut est COMPOUND. COMPOUND est recommandé lorsque vous mettez régulièrement à jour vos tables avec des opérations INSERT, UPDATE ou DELETE. Une clé de tri INTERLEAVED peut utiliser huit colonnes au maximum. Selon la taille de vos données et de votre cluster, l'exécution de VACUUM REINDEX prend beaucoup plus de temps que celle de VACUUM FULL car l'opération effectue un passage supplémentaire pour analyser les clés de tri entrelacées. L'opération de tri et de fusion peut prendre plus de temps pour les tables entrelacées, car le tri entrelacé peut nécessiter la réorganisation de plus de lignes qu'un tri composé.

Pour afficher les clés de tri d'une table, interrogez la vue système SVV_TABLE_INFO.

Rubriques

- Tri multidimensionnel de la disposition des données (version préliminaire)

- Clé de tri composée
- Clé de tri entrelacée

Tri multidimensionnel de la disposition des données (version préliminaire)

Ceci est une version préliminaire de la documentation sur le tri multidimensionnel de la dispositi on des données des tables, qui est en version préliminaire. La documentation et la fonction sont toutes deux sujettes à modification. Nous vous recommandons d'utiliser cette fonction uniquemen t avec des clusters de test et non dans des environnements de production. Pour connaître les conditions générales de la version préliminaire, veuillez consulter la rubrique Participation au service Bêta dans les Conditions générales du service AWS.

> **Note**
>
> Cette fonctionnalité n'est disponible qu'à l'aide d'un cluster de prévisualisation. Pour créer un cluster en version préliminaire, consultez Création d'un cluster en version préliminaire dans le Guide de gestion Amazon Redshift.

Une clé de tri multidimensionnel de la disposition des données est un type de clé de tri AUTO basée sur des prédicats répétitifs trouvés dans une charge de travail. Si votre charge de travail comporte des prédicats répétitifs, Amazon Redshift peut améliorer les performances d'analyse des tables en colocalisant les lignes de données qui répondent aux prédicats répétitifs. Au lieu de stocker les données d'une table dans un ordre de colonnes strict, une clé de tri multidimensionnel de la disposition des données stocke les données en analysant les prédicats répétitifs qui apparaissent dans une charge de travail. Plusieurs prédicats répétitifs peuvent être trouvés dans une charge de travail. En fonction de la charge de travail, ce type de clé de tri peut améliorer les performances de nombreux prédicats. Amazon Redshift détermine automatiquement si cette méthode de clé de tri doit être utilisée pour les tables définies avec une clé de tri `AUTO`.

Par exemple, supposons que vous disposiez d'un jeu de données trié par ordre de colonne. Il peut être nécessaire d'examiner de nombreux blocs de données pour déterminer s'ils répondent aux prédicats de la charge de travail. Toutefois, si les données sont stockées sur le disque dans un ordre de prédicat, moins de blocs doivent être analysés pour répondre à la requête. Dans ce cas, l'utilisation d'une clé de tri multidimensionnel de la disposition des données est avantageuse.

Pour savoir si une requête utilise une clé de mise en page des données multidimensionnelle, consultez la colonne `step_attribute` de la vue SYS_QUERY_DETAIL. Lorsque la valeur est `multi-dimensional`, la disposition multidimensionnelle des données a été utilisée pour la requête. Pour savoir si une requête utilise une clé de tri multidimensionnel de la disposition des données, consultez la colonne `sortkey1` de la vue SVV_TABLE_INFO. Lorsque la valeur est `padb_internal_mddl_key_col`, la disposition multidimensionnelle des données a été utilisée pour la clé de tri de la table.

Pour empêcher Amazon Redshift d'utiliser une clé de tri multidimensionnel de la disposition des données, choisissez une option de clé de tri de table autre que `SORTKEY AUTO`. Pour plus d'informations sur les options SORTKEY, consultez CREATE TABLE.

Clé de tri composée

Une clé de tri composée est constituée de toutes les colonnes répertoriées dans la définition de clé de tri, dans leur ordre d'apparition. Une clé de tri composée est très utile lorsqu'un filtre de requête applique des conditions, telles que des filtres et des jointures, qui utilisent un préfixe de clés de tri. Les avantages du tri composé en termes de performances diminuent lorsque les requêtes dépendent seulement de colonnes de tri secondaires, sans référence aux colonnes principales. COMPOUND est le type de tri par défaut.

Les clés de tri composées peuvent augmenter la vitesse des jointures, les opérations GROUP BY et ORDER BY et les fonctions de fenêtrage qui utilisent PARTITION BY et ORDER BY. Par exemple, une jointure par fusion, ce qui est souvent plus rapide qu'une jointure par hachage, est possible lorsque les données sont distribuées et pré-triées sur les colonnes de jointure. Les clés de tri composées permettent également d'améliorer la compression.

Lorsque vous ajoutez des lignes à une table triée contenant déjà des données, la région non triée se développe, ce qui a une incidence significative sur les performances. C'est encore plus flagrant lorsque la table utilise le tri entrelacé, en particulier lorsque les colonnes de tri incluent des données qui augmentent de façon uniforme, telles que les colonnes de date ou d'horodatage. Exécutez régulièrement une opération VACUUM, surtout après des charges de données volumineuses, pour trier et analyser à nouveau les données. Pour de plus amples informations, veuillez consulter Réduire la taille de la région non triée. Après avoir effectué une opération « vacuum » pour trier à nouveau les données, il est recommandé d'exécuter une commande ANALYZE pour mettre à jour les métadonnées statistiques pour le planificateur de requête. Pour plus d'informations, consultez Analyse des tables.

Clé de tri entrelacée

Un tri entrelacé confère un poids égal à chaque colonne, ou sous-ensemble de colonnes, de la clé de tri. Si plusieurs requêtes utilisent des colonnes différentes pour les filtres, vous pouvez souvent améliorer les performances de ces requêtes en utilisant un style de tri entrelacé. Quand une requête utilise des prédicats restrictifs sur les colonnes de tri secondaires, le tri entrelacé améliore considérablement les performances des requêtes par rapport à un tri composé.

> **Important**
>
> N'utilisez pas de clé de tri entrelacée sur les colonnes qui contiennent des attributs qui augmentent de façon monotone, tels que les colonnes d'identité, les dates ou les horodatages.

L'amélioration des performances que vous obtenez en mettant en œuvre une clé de tri entrelacée doit être comparée aux temps de charge et de vidage.

Les tris entrelacés sont plus efficaces avec des requêtes extrêmement sélectives qui filtrent une ou plusieurs colonnes de clés dans la clause WHERE, par exemple `select c_name from customer where c_region = 'ASIA'`. Les avantages du tri entrelacé augmentent avec le nombre de colonnes triées restreintes.

Un tri entrelacé est plus efficace avec des tables volumineuses. Le tri est appliqué sur chaque tranche. Ainsi, un tri entrelacé est plus efficace lorsqu'une table est suffisamment volumineuse pour nécessiter plusieurs blocs de 1 Mo par tranche. Ici, le processeur de requêtes peut ignorer une proportion significative des blocs en utilisant des prédicats restrictifs. Pour afficher le nombre de blocs utilisés par une table, interrogez la vue système STV_BLOCKLIST.

Lorsque vous effectuez un tri sur une seule colonne, un tri entrelacé peut offrir de meilleures performances qu'un tri composé si les valeurs de la colonne ont un préfixe de commun. Par exemple, commencez URLs généralement par « http://www ». Les clés de tri composées utilisent un nombre limité de caractères après le préfixe, ce qui se traduit par un grand nombre de duplications de clés. Les tris entrelacés utilisent un schéma de compression interne pour les valeurs de la carte de zone qui leur permet de mieux distinguer les valeurs de la colonne ayant un préfixe commun long.

Lors de la migration de clusters provisionnés Amazon Redshift vers Amazon Redshift sans serveur, Redshift convertit les tables comportant des clés de tri entrelacées et DISTSTYLE KEY en clés de

tri composées. Le DISTSTYLE ne change pas. Pour en savoir plus sur les styles de distribution, consultez Utilisation des styles de distribution de données.

VACUUM REINDEX

Lorsque vous ajoutez des lignes à une table triée contenant déjà des données, les performances peuvent se détériorer au fil du temps. Cette détérioration se produit pour les tris composés et entrelacés, mais elle a des conséquences plus graves sur les tables. Une commande VACUUM permet de restaurer l'ordre de tri, mais l'opération peut prendre plus de temps pour les tables entrelacées, car la fusion de nouvelles données entrelacées peut impliquer la modification de chaque bloc de données.

Lorsque les tables sont initialement chargées, Amazon Redshift analyse la distribution des valeurs dans les colonnes de clés de tri et utilise ces informations pour un entrelacement optimal des colonnes de clés. A mesure qu'une table se développe, la distribution des valeurs dans les colonnes de clé de tri peut changer, ou se décaler, en particulier avec les colonnes de date ou d'horodatage. Si l'asymétrie devient trop grande, les performances peuvent en être affectées. Pour analyser à nouveau les clés de tri et restaurer les performances, exécutez la commande VACUUM avec le mot clé REINDEX. Du fait qu'il a besoin d'effectuer un passage d'analyse des données supplémentaire, VACUUM REINDEX peut prendre plus de temps qu'une commande VACUUM standard pour les tables entrelacées. Pour afficher des informations sur l'asymétrie de la distribution de clés et la durée de la dernière réindexation, interrogez la vue système SVV_INTERLEAVED_COLUMNS.

Pour plus d'informations sur la façon de déterminer la fréquence d'exécution de VACUUM et sur le moment d'exécution d'un VACUUM REINDEX, consultez Décidez s'il faut réindexer.

Contraintes liées aux tables

Les contraintes d'unicité, de clé primaire et de clé externe sont uniquement informatives ; elles ne sont pas appliquées par Amazon Redshift lorsque vous remplissez une table. Par exemple, si vous insérez des données dans une table avec des dépendances, l'insertion peut réussir même si elle enfreint la contrainte. Néanmoins, les clés primaires et les clés étrangères servent de conseils de planification et doivent être déclarées si votre processus ETL ou un autre processus de votre application impose leur intégrité.

Par exemple, le gestionnaire de requêtes utilise des clés primaires et étrangères dans certains calculs statistiques. Il le fait pour déduire l'unicité et les relations référentielles qui affectent les techniques de décorrélation des sous-requêtes. Ce faisant, il peut ordonner un grand nombre de jointures et supprimer les jointures redondantes.

Le planificateur exploite ces relations entre les clés, mais il suppose que toutes les clés des tables Amazon Redshift sont valides telles qu'elles sont chargées. Si votre application autorise les clés étrangères ou primaires non valides, certaines requêtes peuvent renvoyer des résultats incorrects. Par exemple, une requête SELECT DISTINCT peut renvoyer des lignes en double si la clé primaire n'est pas unique. Ne définissez pas de contraintes de clés pour vos tables si vous doutez de leur validité. Cependant, déclarez toujours les clés primaires et étrangères et les contraintes d'unicité lorsque vous savez qu'elles sont valides.

Amazon Redshift fait respecter les contraintes de colonne NOT NULL.

Pour de plus amples informations sur les contraintes de table, consultez CREATE TABLE. Pour plus d'informations sur la manière de supprimer une table avec des dépendances, consultez DROP TABLE.

Chargement de données dans Amazon Redshift

Il existe plusieurs méthodes pour charger des données dans une base de données Amazon Redshift. Les fichiers Amazon S3 sont l'une des sources de données les plus populaires à charger. Le tableau suivant récapitule certaines des méthodes à utiliser pour démarrer à partir d'une source Amazon S3.

Méthode à utiliser	Description	Quand la méthode est nécessaire
COPY command (Commande COPY)	Exécute une ingestion de fichiers par lots pour charger des données à partir de vos fichiers Amazon S3. Cette méthode tire parti des capacités de traitement parallèle d'Amazon Redshift. Pour de plus amples informations, veuillez consulter Chargement de tables à l'aide de la commande COPY.	Doit être utilisé lorsque les exigences de base en matière de chargement de données pour initier manuellement l'ingestion de fichiers par lots sont nécessaires. Cette méthode est principalement utilisée avec des pipelines d'ingestion de fichiers personnalisés et tiers ou des charges de travail d'ingestion de fichiers ponctuelles ou ad hoc.
COPIE... Commande CREATE JOB (copie automatique)	Exécute automatiquement vos commandes COPY lorsqu'un nouveau fichier est créé sur des chemins Amazon S3 suivis. Pour de plus amples informations, veuillez consulter Créez une intégration d'événements S3 pour copier automatiquement des fichiers depuis des compartiments Amazon S3.	À utiliser lorsqu'un pipeline d'ingestion de fichiers doit automatiquement ingérer des données lorsqu'un nouveau fichier est créé sur Amazon S3. Amazon Redshift assure le suivi des fichiers ingérés afin d'éviter la duplication des données. Cette méthode nécessite une configuration par les propriétaires de compartiments Amazon S3.

Méthode à utiliser	Description	Quand la méthode est nécessaire
Charger des requêtes à partir d'un lac de données	Créez des tables externes pour exécuter des requêtes de lac de données sur vos fichiers Amazon S3, puis exécutez la commande INSERT INTO pour charger les résultats de vos requêtes de lac de données dans des tables locales. Pour de plus amples informations, veuillez consulter Tables externes pour Redshift Spectrum.	Doit être utilisé dans l'un des scénarios suivants : • Chargement depuis AWS Glue et ouverture de formats de table (tels qu'Apache Iceberg, Apache Hudi ou Delta Lake). • Les fichiers source doivent être partiellement ingérés (par exemple, nécessaires pour exécuter une clause WHERE pour ingérer des lignes particulières). • Plus de flexibilité est nécessaire pour ingérer des colonnes spécifiques (par exemple, exécuter une commande SELECT) ou effectuer des transform ations de données de base en déplacement (par exemple, appliquer des opérations de base ou appeler UDFs les valeurs du fichier source).
Autres méthodes que vous pouvez envisager		

Méthode à utiliser	Description	Quand la méthode est nécessaire
Ingestion en streaming	L'ingestion du streaming permet une ingestion rapide et à faible latence des données de flux provenant d'Amazon Kinesis Data Streams et d'Amazon Managed Streaming for Apache Kafka dans une vue matérialisée provisionnée par Amazon Redshift ou Redshift Serverless. Pour plus d’informations, consultez Mise en route de l’ingestion en streaming à partir d’Amazon Kinesis Data Streams et Commencer à ingérer du streaming à partir de sources Apache Kafka.	À prendre en compte pour les cas d'utilisation où les données sont d'abord diffusées dans des fichiers sur Amazon S3, puis chargées depuis Amazon S3. S'il n'est pas nécessaire de conserver des données sur Amazon S3, vous pouvez souvent envisager de les diffuser directement dans Amazon Redshift.
Exécution de requêtes sur le lac de données	Exécution de requêtes directement depuis une table de lac de données au lieu d'ingérer le contenu de la table dans une table locale. Pour de plus amples informations, veuillez consulter Amazon Redshift Spectrum.	À utiliser lorsque le cas d'utilisation ne nécessite pas les performances des requêtes de table locales dans Amazon Redshift.

Méthode à utiliser	Description	Quand la méthode est nécessaire
Chargement par lots à l'aide de l'éditeur de requêtes Amazon Redshift v2	Vous pouvez préparer et exécuter visuellement vos charges de travail d'ingestion de fichiers par lots dans l'éditeur de requêtes Amazon Redshift v2. Pour plus d'informations, consultez la section Chargement de données depuis S3 dans le guide de gestion Amazon Redshift.	À utiliser lorsque vous souhaitez que l'éditeur de requêtes v2 prépare des instructions COPY et que vous souhaitez un outil visuel pour simplifier le processus de préparation des instructions COPY.
Charger des données depuis un fichier local à l'aide de l'éditeur de requêtes Amazon Redshift v2	Vous pouvez charger des fichiers directement depuis votre bureau vers les tables Amazon Redshift sans avoir à les télécharger manuellement dans Amazon S3. Pour plus d'informations, consultez la section Chargement de données à partir d'une configuration de fichiers et d'un flux de travail locaux dans le guide de gestion Amazon Redshift.	À utiliser lorsque vous devez charger rapidement des fichiers depuis votre ordinateur local à des fins de requêtes ponctuelles. Avec cette méthode, l'éditeur de requêtes Amazon Redshift v2 stocke temporairement le fichier dans un compartiment Amazon S3 appartenant au client et exécute une commande de copie en utilisant ce chemin Amazon S3.

Une commande COPY est le moyen le plus efficace de charger une table. Vous pouvez également ajouter des données à vos tables à l’aide des commandes INSERT, même si c’est beaucoup moins efficace que l’utilisation de la commande COPY. La commande COPY est capable de lire simultanément plusieurs fichiers de données ou plusieurs flux de données. Amazon Redshift alloue la charge de travail aux nœuds Amazon Redshift et effectue les opérations de chargement en parallèle, notamment en triant les lignes et en distribuant les données entre les tranches de nœuds.

> **Note**
>
> Les tables externes d'Amazon Redshift Spectrum sont en lecture seule. Vous ne pouvez pas copier (COPY) ni insérer (INSERT) dans une table externe.

Pour accéder aux données d'autres AWS ressources, Amazon Redshift doit être autorisé à accéder à ces ressources et à effectuer les actions nécessaires pour accéder aux données. Vous pouvez utiliser AWS Identity and Access Management (IAM) pour limiter l'accès des utilisateurs aux ressources et aux données Amazon Redshift.

Une fois vos données initiales chargées, si vous ajoutez, modifiez ou supprimez une quantité importante de données, vous devez poursuivre en exécutant une commande VACUUM pour réorganiser vos données et récupérer de l'espace après les suppressions. Vous devez également exécuter une commande ANALYZE pour mettre à jour les statistiques des tables.

Rubriques

- Chargement de tables à l'aide de la commande COPY
- Créez une intégration d'événements S3 pour copier automatiquement des fichiers depuis des compartiments Amazon S3
- Chargement de tables à l'aide de commandes DML
- Exécution d'une copie complète
- Analyse des tables
- Exécution de l'opération VACUUM sur les tables
- Gestion des opérations d'écriture simultanées
- Didacticiel : chargement des données à partir d'Amazon S3

Chargement de tables à l'aide de la commande COPY

La commande COPY s'appuie sur l'architecture Amazon Redshift du traitement hautement parallèle (MPP) pour lire et charger les données en parallèle à partir de fichiers sur Amazon S3, d'une table DynamoDB ou de la sortie de texte d'un ou de plusieurs hôtes distants.

Avant de connaître toutes les options de la commande COPY, nous vous recommandons de connaître les options de base pour charger les données Amazon S3. Le guide de démarrage Amazon

Redshift montre une utilisation simple de la commande COPY pour charger des données Amazon S3 à l'aide d'un rôle IAM par défaut. Consultez Étape 4 : Charger des données depuis Amazon S3 vers Amazon Redshift pour plus de détails.

> **Note**
>
> Nous vous recommandons vivement d'utiliser la commande COPY pour charger de grandes quantités de données. La lenteur liée à l'utilisation d'instructions INSERT pour remplir une table peut être prohibitive. Sinon, si vos données existent déjà dans d'autres tables de bases de données Amazon Redshift, utilisez INSERT INTO ... SELECT ou CREATE TABLE AS pour améliorer les performances. Pour plus d'informations, consultez INSERT ou CREATE TABLE AS.

Pour charger des données depuis une autre AWS ressource, Amazon Redshift doit être autorisé à accéder à la ressource et à effectuer les actions nécessaires.

Pour accorder ou révoquer le privilège de charger les données dans une table à l'aide d'une commande COPY, accordez ou révoquez le privilège INSERT.

Vos données doivent être au format approprié pour le chargement dans votre table Amazon Redshift. Cette section présente les directives de préparation et de vérification de vos données avant le chargement, et de validation d'une instruction COPY avant son exécution.

Pour protéger les informations contenues dans vos fichiers, vous pouvez chiffrer les fichiers de données avant de les charger sur votre compartiment Amazon S3 ; la commande COPY déchiffre les données pendant le chargement. Vous pouvez aussi limiter l'accès à vos données de chargement en fournissant des informations d'identification de sécurité temporaires aux utilisateurs. Les informations d'identification de sécurité temporaires offrent une sécurité améliorée parce qu'elles sont de courte durée et ne peuvent pas être réutilisées après leur expiration.

Amazon Redshift dispose de fonctions intégrées à COPY pour charger rapidement des données délimitées et non compressées. Vous pouvez toutefois compresser vos fichiers à l'aide de gzip, lzop ou bzip2 pour gagner du temps lors du téléchargement des fichiers.

Si les mots clés suivants figurent dans la requête COPY, le fractionnement automatique des données non compressées n'est pas pris en charge : ESCAPE, REMOVEQUOTES et FIXEDWIDTH. Mais le mot-clé CSV est pris en charge.

Pour garantir la sécurité de vos données en transit dans le AWS cloud, Amazon Redshift utilise le protocole SSL à accélération matérielle pour communiquer avec Amazon S3 ou Amazon DynamoDB pour les opérations de copie, de déchargement, de sauvegarde et de restauration.

Lorsque vous chargez directement votre table depuis une table Amazon DynamoDB, vous avez la possibilité de contrôler la quantité de débit provisionné Amazon DynamoDB que vous consommez.

Vous pouvez, le cas échéant, laisser la commande COPY analyser vos données d'entrée et appliquer automatiquement les encodages de compression optimale à votre table dans le cadre du processus de chargement.

Rubriques

- Informations d'identification et autorisations d'accès
- Préparation de vos données d'entrée
- Chargement des données à partir d'Amazon S3
- Chargement de données à partir d'Amazon EMR
- Chargement des données à partir des hôtes distants
- Chargement de données à partir d'une table Amazon DynamoDB
- Vérification que les données ont été chargées correctement
- Validation des données d'entrée
- Chargement des tables avec compression automatique
- Optimisation de stockage pour les tables étroites
- Chargement des valeurs par défaut des colonnes
- Résolution des problèmes de chargement de données

Informations d'identification et autorisations d'accès

Pour charger ou décharger des données à l'aide d'une autre AWS ressource, telle qu'Amazon S3, Amazon DynamoDB, Amazon EMR ou Amazon, EC2 Amazon Redshift doit être autorisé à accéder à la ressource et à effectuer les actions nécessaires pour accéder aux données. Par exemple, pour charger des données depuis Amazon S3, la commande COPY doit disposer de l'accès LIST au compartiment et de l'accès GET pour les objets du compartiment.

Pour obtenir l'autorisation d'accéder à une ressource, Amazon Redshift doit être authentifié. Vous pouvez choisir le contrôle d'accès basé sur les rôles ou le contrôle d'accès basé sur les clés. Cette

section présente une vue d'ensemble des deux méthodes. Pour obtenir des détails et des exemples, consultez Autorisations d'accès aux autres ressources AWS.

Contrôle d'accès basé sur les rôles

Grâce au contrôle d'accès basé sur les rôles, Amazon Redshift assume temporairement AWS Identity and Access Management un rôle (IAM) en votre nom. Ensuite, en fonction des autorisations accordées au rôle, Amazon Redshift peut accéder aux AWS ressources requises.

Nous vous recommandons d'utiliser le contrôle d'accès basé sur les rôles, car il permet un contrôle plus sûr et plus précis de l'accès aux AWS ressources et aux données utilisateur sensibles, en plus de protéger vos informations d'identification. AWS

Pour utiliser le contrôle d'accès basé sur les rôles, vous devez d'abord créer un rôle IAM en utilisant le type de rôle de service Amazon Redshift, puis associer le rôle à votre entrepôt de données. Le rôle doit avoir, au minimum, les autorisations répertoriées dans Autorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY. Pour savoir comment créer un rôle IAM et l'associer à votre cluster, consultez la section Création d'un rôle IAM pour autoriser votre cluster Amazon Redshift à AWS accéder aux services dans le guide de gestion Amazon Redshift.

Vous pouvez ajouter un rôle à un cluster ou afficher les rôles associés à un cluster à l'aide de la console de gestion, de la CLI ou de l'API Amazon Redshift. Pour plus d'informations, consultez Authorizing COPY and UNLOAD Operations Using IAM Roles dans le Guide de gestion Amazon Redshift.

Lorsque vous créez un rôle IAM, IAM renvoie un Amazon Resource Name (ARN) pour le rôle. Pour exécuter une commande COPY à l'aide d'un rôle IAM, fournissez le rôle ARN à l'aide des paramètres IAM_ROLE ou CREDENTIALS.

L'exemple suivant de commande COPY utilise le paramètre IAM_ROLE avec le rôle `MyRedshiftRole` pour l'authentification.

```
COPY customer FROM 's3://amzn-s3-demo-bucket/mydata'
IAM_ROLE 'arn:aws:iam::12345678901:role/MyRedshiftRole';
```

L' AWS utilisateur doit disposer, au minimum, des autorisations répertoriées dansAutorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY.

Contrôle d’accès basé sur les clés

Avec le contrôle d'accès basé sur les clés, vous fournissez l'ID de clé d'accès et la clé d'accès secrète à un utilisateur autorisé à accéder aux AWS ressources contenant les données.

> **Note**
>
> Nous vous recommandons vivement d’utiliser un rôle IAM pour l’authentification au lieu de fournir une clé d’accès secrète et un ID de clé d’accès en texte brut. Si vous optez pour le contrôle d'accès par clé, n'utilisez jamais les informations d'identification de votre AWS compte (root). Créez toujours un utilisateur IAM et fournissez l’ID de clé d’accès et la clé d’accès secrète de cet utilisateur. Pour connaître les étapes à suivre pour créer un utilisateur IAM, consultez Création d’un utilisateur IAM dans votre compte AWS.

Préparation de vos données d’entrée

Si vos données d’entrée ne sont pas compatibles avec les Colonnes de la table qui les recevront, la commande COPY échoue.

Utilisez les instructions suivantes pour vous assurer que vos données d’entrée sont valides :

- Vos données peuvent contenir uniquement des caractères UTF-8 de quatre octets au plus.
- Vérifiez que les chaînes CHAR et VARCHAR ne sont pas plus longues que les colonnes correspondantes. Comme les chaînes VARCHAR sont mesurées en octets, pas en caractères, une chaîne de quatre caractères chinois qui occupent quatre octets chacun nécessite une colonne VARCHAR(16), par exemple.
- Les caractères multioctets ne peuvent être utilisés qu’avec les colonnes VARCHAR. Vérifiez que les caractères multioctets ne dépassent pas quatre octets.
- Vérifiez que les données des colonnes CHAR ne contiennent que des caractères codés sur un octet.
- N’incluez pas de caractères spéciaux ou de syntaxe particulière pour indiquer le dernier champ d’un enregistrement. Ce champ peut être un délimiteur.
- Si vos données incluent des terminaisons null, également appelées NUL (UTF-8 0000) ou zéro binaire (0x000), vous pouvez charger ces caractères en tant que valeurs NULL dans des colonnes CHAR ou VARCHAR en utilisant l’option NULL AS dans la commande COPY : `null as '\0'` ou

`null as '\000'`. Si vous n'utilisez pas NULL AS, les terminaisons null entraînent l'échec de la commande COPY.

- Si vos chaînes contiennent des caractères spéciaux, tels que des délimiteurs et des sauts de ligne imbriqués, utilisez l'option ESCAPE avec la commande COPY.
- Vérifiez que tous les guillemets simples et doubles correspondent de manière appropriée.
- Vérifiez que les chaînes à virgule flottante sont dans un format à virgule flottante standard, tel que 12,123, ou dans un format d'élévation à la puissance, tel que 1,0E4.
- Vérifiez que toutes les chaînes de type timestamp et date suivent les spécifications de Chaînes DATEFORMAT et TIMEFORMAT. Le format d'horodatage par défaut est YYYY-MM-DD hh:mm:ss, et le format de date par défaut est. YYYY-MM-DD
- Pour plus d'informations sur les limites et limitations des types de données individuels, consultez Types de données. Pour plus d'informations sur les erreurs de caractères multioctets, consultez Erreurs de chargement de caractères multioctets

Chargement des données à partir d'Amazon S3

La commande COPY s'appuie sur l'architecture Amazon Redshift du traitement hautement parallèle (MPP) pour lire et charger les données en parallèle à partir d'un ou de plusieurs fichiers d'un compartiment Amazon S3. Vous pouvez tirer le meilleur parti du traitement parallèle en fractionnant vos données en plusieurs fichiers, dans les cas où les fichiers sont compressés. (Il existe des exceptions à cette règle, qui sont détaillées dans la section Chargement de fichiers de données.) Vous pouvez aussi profiter au maximum du traitement parallèle en définissant les clés de distribution de vos tables. Pour plus d'informations sur les clés de distribution, consultez Distribution des données pour l'optimisation des requêtes.

Les données sont chargées dans la table cible, une ligne par ligne. Les champs du fichier de données sont mis en correspondance avec les colonnes de la table dans l'ordre, de gauche à droite. Les champs des fichiers de données peuvent être à largeur fixe ou délimités par un caractère ; le délimiteur par défaut est une barre verticale (|). Par défaut, toutes les colonnes de la table sont chargées, mais vous pouvez, le cas échéant, définir une liste de colonnes séparées par des virgules. Si une colonne de table n'est pas incluse dans la liste des colonnes spécifiée dans la commande COPY, elle est chargée avec une valeur par défaut. Pour plus d'informations, consultez Chargement des valeurs par défaut des colonnes.

Rubriques

- Chargement de données à partir de fichiers compressés et non compressés

- Téléchargement de fichiers vers Amazon S3 à utiliser avec COPY
- Utilisation de la commande COPY pour charger à partir d'Amazon S3

Chargement de données à partir de fichiers compressés et non compressés

Lorsque vous chargez des données compressées, nous vous recommandons de fractionner les données de chaque table en plusieurs fichiers. Lorsque vous chargez des données délimitées et non compressées, la commande COPY utilise des plages de traitement massivement parallèle (MPP) et d'analyse pour charger des données à partir de fichiers volumineux dans un compartiment Amazon S3.

Chargement de données à partir de plusieurs fichiers compressés

Dans les cas où vous avez des données compressées, nous vous recommandons de fractionner les données de chaque table en plusieurs fichiers. La commande COPY peut charger les données à partir de plusieurs fichiers en parallèle. Vous pouvez charger plusieurs fichiers en spécifiant un préfixe commun, ou clé de préfixe, pour l'ensemble, ou en indiquant explicitement les fichiers dans un fichier manifeste.

Scindez vos données en fichiers de telle sorte que le nombre de fichiers soit un multiple du nombre de tranches de votre cluster. De cette façon, Amazon Redshift peut diviser également les données entre les tranches. Le nombre de tranches par nœud dépend de la taille de nœud du cluster. Par exemple, chaque nœud de calcul dc2.large possède deux tranches, et chaque nœud de calcul dc2.8xlarge possède 16 tranches. Pour plus d'informations sur le nombre de tranches pour chaque taille de nœud, consultez la rubrique À propos des clusters et nœuds dans le Guide de gestion Amazon Redshift.

Les nœuds participent tous à l'exécution de requêtes parallèles, en travaillant sur des données réparties aussi uniformément que possible sur les tranches. Si vous avez un cluster avec deux nœuds dc2.large, vous pouvez diviser vos données en quatre fichiers ou un multiple de quatre. Amazon Redshift ne prend pas en compte la taille des fichiers lors de la répartition de la charge de travail. Par conséquent, vous devez vous assurer que les fichiers ont à peu près la même taille, de 1 Mo à 1 Go après compression.

Pour utiliser des préfixes d'objet pour identifier les fichiers du chargement, nommez chaque fichier avec un préfixe commun. Par exemple, le fichier `venue.txt` peut être fractionné en quatre fichiers, comme suit :

```
venue.txt.1
```

```
venue.txt.2
venue.txt.3
venue.txt.4
```

Si vous placez plusieurs fichiers dans un dossier de votre compartiment et spécifiez le nom du dossier comme préfixe, la commande COPY charge tous les fichiers dans le dossier. Si vous listez explicitement les fichiers à charger à l'aide d'un fichier manifeste, les fichiers peuvent résider dans différents compartiments ou dossiers.

Pour plus d'informations sur les fichiers manifeste, consultez Example: COPY from Amazon S3 using a manifest.

Chargement de données à partir de fichiers délimités non compressés

Lorsque vous chargez des données non compressées et délimitées, la commande COPY utilise l'architecture de traitement massivement parallèle (MPP) dans Amazon Redshift. Amazon Redshift utilise automatiquement des tranches fonctionnant en parallèle avec des plages de chargement de données provenant d'un fichier volumineux dans un compartiment Amazon S3. Le fichier doit être délimité pour que le chargement en parallèle ait lieu. Par exemple, délimitation de canaux. Le chargement automatique et parallèle des données avec la commande COPY est également disponible pour les fichiers CSV. Vous pouvez également profiter du traitement parallèle en définissant les clés de distribution de vos tables. Pour plus d'informations sur les clés de distribution, consultez Distribution des données pour l'optimisation des requêtes.

Le chargement parallèle des données n'est pas pris en charge lorsque la requête COPY inclut l'un des mots clés suivants : ESCAPE, REMOVEQUOTES et FIXEDWIDTH.

Les données du ou des fichiers sont chargées dans la table cible, une ligne par ligne. Les champs du fichier de données sont mis en correspondance avec les colonnes de la table dans l'ordre, de gauche à droite. Les champs des fichiers de données peuvent être à largeur fixe ou délimités par un caractère ; le délimiteur par défaut est une barre verticale (|). Par défaut, toutes les colonnes de la table sont chargées, mais vous pouvez, le cas échéant, définir une liste de colonnes séparées par des virgules. Si une colonne de table n'est pas incluse dans la liste des colonnes spécifiée dans la commande COPY, elle est chargée avec une valeur par défaut. Pour plus d'informations, consultez Chargement des valeurs par défaut des colonnes.

Suivez ce processus général pour charger les données depuis Amazon S3, lorsque vos données sont décompressées et délimitées :

1. Chargez vos fichiers sur Amazon S3.

2. Exécutez une commande COPY pour charger la table.
3. Vérifiez que les données ont été chargées correctement.

Pour obtenir des exemples de commandes COPY, consultez Exemples de commandes COPY. Pour plus d'informations sur les données chargées dans Amazon Redshift, consultez les tables système STL_LOAD_COMMITS et STL_LOAD_ERRORS.

Pour plus d'informations sur les nœuds et les tranches qu'ils contiennent, consultez À propos des clusters et nœuds dans le Guide de gestion Amazon Redshift.

Téléchargement de fichiers vers Amazon S3 à utiliser avec COPY

Il existe quelques approches à adopter lors du chargement de fichiers texte sur Amazon S3 :

- Si vous avez des fichiers compressés, nous vous recommandons de fractionner les fichiers volumineux pour tirer parti du traitement parallèle dans Amazon Redshift.
- D'autre part, COPY divise automatiquement les données de fichiers volumineux, non compressés et délimités par des textes, afin de faciliter le parallélisme et de distribuer efficacement les données de fichiers volumineux.

Créez un compartiment Amazon S3 pour contenir vos fichiers de données, puis chargez les fichiers de données dans le compartiment. Pour plus d'informations sur la création de compartiments et le chargement de fichiers, consultez Working with Amazon S3 Buckets dans le Guide de l'utilisateur Amazon Simple Storage Service.

> **Important**
>
> Le compartiment Amazon S3 qui contient les fichiers de données doit être créé dans la même région AWS que votre cluster, sauf si vous utilisez l'option REGION pour spécifier la région dans laquelle se trouve le compartiment Amazon S3.

Vérifiez que les plages d'adresses IP S3 sont ajoutées à votre liste d'autorisations. Pour plus d'informations sur les plages d'adresses IP S3 requises, consultez Isolement de réseau.

Vous pouvez créer un compartiment Amazon S3 dans une région spécifique en sélectionnant la région lorsque vous créez le compartiment à l'aide de la console Amazon S3 ou en spécifiant un point de terminaison lorsque vous créez le compartiment à l'aide de l'API ou de la CLI Amazon S3.

Après le chargement des données, vérifiez que les fichiers appropriés sont présents sur Amazon S3.

Rubriques

- Gestion de la cohérence des données
- Chargement de données chiffrées sur Amazon S3
- Vérification de la présence des fichiers corrects dans votre compartiment

Gestion de la cohérence des données

Amazon S3 assure une forte read-after-write cohérence pour les opérations COPY, UNLOAD, INSERT (table externe), CREATE EXTERNAL TABLE AS et Amazon Redshift Spectrum sur les compartiments Amazon S3 dans toutes les régions. AWS En outre, les opérations de lecture sur Amazon S3 Select, les listes de contrôle d'accès Amazon S3, les balises d'objet Amazon S3 et les métadonnées d'objet (par exemple, l'objet HEAD) sont fortement cohérentes. Pour plus d'informations sur la cohérence des données, consultez Modèle de cohérence des données Amazon S3 dans le Guide de l'utilisateur Amazon Simple Storage Service.

Chargement de données chiffrées sur Amazon S3

Amazon S3 prend en charge le chiffrement côté serveur et le chiffrement côté client. Cette rubrique présente les différences entre le chiffrement côté serveur et le chiffrement côté client, et décrit les étapes pour utiliser le chiffrement côté client avec Amazon Redshift. Le chiffrement côté serveur est transparent à Amazon Redshift.

Chiffrement côté serveur

Le chiffrement côté serveur est le chiffrement des données au repos : autrement dit, Amazon S3 chiffre vos données au fur et à mesure qu'il les charge et les déchiffre automatiquement lorsque vous y accédez. Lorsque vous chargez les tables à l'aide d'une commande COPY, il n'existe aucune différence dans la façon dont vous chargez à partir d'objets non chiffrés ou d'objets chiffrés côté serveur sur Amazon S3. Pour plus d'informations sur le chiffrement côté serveur, consultez Using Server-Side Encryption dans le Guide de l'utilisateur Amazon Simple Storage Service.

Chiffrement côté client

Dans le chiffrement côté client, votre application cliente gère le chiffrement de vos données, les clés de chiffrement et les outils associés. Vous pouvez charger les données sur un compartiment Amazon S3 à l'aide d'un chiffrement côté client, puis chargez les données à l'aide de la commande COPY avec l'option ENCRYPTED et une clé de chiffrement privée pour renforcer la sécurité.

Vous chiffrez vos données à l'aide du chiffrement d'enveloppe. Avec le chiffrement d'enveloppe, votre application gère tout le chiffrement exclusivement. Vos clés de chiffrement privées et vos données non chiffrées ne sont jamais envoyées à AWS. Il est donc très important que vous gériez vos clés de chiffrement en toute sécurité. Si vous perdez vos clés de chiffrement, vous ne pourrez pas déchiffrer vos données et vous ne pourrez pas récupérer vos clés de chiffrement. AWS Le chiffrement d'enveloppe combine les performances du chiffrement symétrique rapide tout en assurant la plus grande sécurité que la gestion des clés avec des clés asymétriques fournit. Une clé one-time-use symétrique (la clé symétrique d'enveloppe) est générée par votre client de chiffrement Amazon S3 pour chiffrer vos données, puis cette clé est chiffrée par votre clé racine et stockée avec vos données dans Amazon S3. Quand Amazon Redshift accède à vos données pendant un chargement, la clé symétrique chiffrée est récupérée et déchiffrée avec votre clé réelle, puis les données sont déchiffrées.

Pour utiliser les données chiffrées côté client Amazon S3 dans Amazon Redshift, suivez les étapes présentées dans la section Protection des données à l'aide du chiffrement côté client dans le Guide de l'utilisateur Amazon Simple Storage Service avec les conditions requises supplémentaires que vous utilisez :

- Chiffrement symétrique : le kit SDK AWS pour la classe Java `AmazonS3EncryptionClient` utilise le chiffrement d'enveloppe, décrit précédemment, qui repose sur le chiffrement à clé symétrique. Utilisez cette classe pour créer un client Amazon S3 et charger les données chiffrées côté client.
- Clé symétrique racine AES 256 bits : une clé racine chiffre la clé d'enveloppe. Vous passez la clé racine à votre instance de la classe `AmazonS3EncryptionClient`. Enregistrez cette clé, car vous en aurez besoin pour copier les données dans Amazon Redshift.
- Métadonnées objet pour stocker la clé d'enveloppe chiffrée : par défaut, Amazon S3 stocke la clé d'enveloppe comme métadonnées objet pour la classe `AmazonS3EncryptionClient`. La clé d'enveloppe chiffrée qui est stockée sous forme de métadonnées objet est utilisée pendant le processus de déchiffrement.

> **Note**
>
> Si vous recevez un message d'erreur de chiffrement lorsque vous utilisez l'API de chiffrement pour la première fois, il se peut que votre version JDK possède un fichier de politique de juridiction Java Cryptography Extension (JCE) qui limite la longueur maximale de la clé pour les transformations de chiffrement et de déchiffrement vers 128 bits. Pour plus d'informations sur la résolution de ce problème, consultez la section Spécification du chiffrement côté client

à l'aide du AWS SDK pour Java dans le guide de l'utilisateur d'Amazon Simple Storage Service.

Pour plus d'informations sur le chargement de fichiers chiffrés côté client dans vos tables Amazon Redshift à l'aide de la commande COPY, consultez Chargement de fichiers de données chiffrés à partir d'Amazon S3.

Exemple : Chargement des données chiffrées côté client

Pour un exemple d'utilisation du AWS SDK pour Java afin de télécharger des données chiffrées côté client, consultez la section Protection des données à l'aide du chiffrement côté client dans le guide de l'utilisateur d'Amazon Simple Storage Service.

La seconde option montre les choix que vous devez faire pendant le chiffrement côté client afin que les données puissent être chargées dans Amazon Redshift. Plus précisément, l'exemple montre l'utilisation de métadonnées objet pour stocker la clé enveloppe chiffrée et l'utilisation d'une clé symétrique racine AES 256 bits.

Cet exemple fournit un exemple de code utilisant le AWS SDK for Java pour créer une clé racine symétrique AES 256 bits et l'enregistrer dans un fichier. L'exemple charge ensuite un objet sur Amazon S3 à l'aide du client de chiffrement S3 qui chiffre d'abord des exemples de données côté client. Cet exemple télécharge également l'objet et vérifie que les données sont identiques.

Vérification de la présence des fichiers corrects dans votre compartiment

Une fois que vous avez chargé vos fichiers sur votre compartiment Amazon S3, nous vous recommandons d'afficher le contenu du compartiment pour vérifier que tous les fichiers appropriés sont présents et qu'aucun fichier indésirable ne s'y trouve. Par exemple, si le compartiment `amzn-s3-demo-bucket` contient un fichier nommé `venue.txt.back`, ce fichier est chargé, peut-être involontairement, par la commande suivante :

```
COPY venue FROM 's3://amzn-s3-demo-bucket/venue' … ;
```

Si vous souhaitez contrôler plus précisément quels fichiers sont chargés, vous pouvez utiliser un fichier manifeste pour répertorier explicitement les fichiers de données. Pour plus d'informations sur l'utilisation d'un fichier manifeste, consultez l'option copy_from_s3_manifest_file de la commande COPY et Example: COPY from Amazon S3 using a manifest dans les exemples de COPY.

Pour plus d'informations sur l'affichage du contenu du compartiment, consultez Affichage des clés d'objet dans le Guide du développeur Amazon S3.

Utilisation de la commande COPY pour charger à partir d'Amazon S3

Utilisez la commande COPY pour charger une table en parallèle à partir de fichiers de données sur Amazon S3. Vous pouvez spécifier les fichiers à charger en utilisant un préfixe d'objet Amazon S3 ou un fichier manifeste.

La syntaxe pour spécifier les fichiers à charger en utilisant un préfixe est la suivante :

```
COPY <table_name> FROM 's3://<bucket_name>/<object_prefix>'
authorization;
```

Le fichier manifeste est un fichier au format JSON qui répertorie les fichiers de données à charger. La syntaxe pour spécifier les fichiers à charger en utilisant un fichier manifeste est la suivante :

```
COPY <table_name> FROM 's3://<bucket_name>/<manifest_file>'
authorization
MANIFEST;
```

La table à charger doit déjà exister dans la base de données. Pour plus d'informations sur la création d'une table, consultez CREATE TABLE dans la référence SQL.

Les valeurs d'autorisation fournissent l' AWS autorisation dont Amazon Redshift a besoin pour accéder aux objets Amazon S3. Pour plus d'informations sur les autorisations requises, consultez Autorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY. La méthode d'authentification recommandée est celle qui consiste à spécifier le paramètre IAM_ROLE et à fournir l'Amazon Resource Name (ARN) d'un rôle IAM avec les autorisations nécessaires. Pour plus d'informations, consultez Contrôle d'accès basé sur les rôles.

Pour vous authentifier à l'aide du paramètre IAM_ROLE, remplacez *<aws-account-id>* et *<role-name>* comme indiqué dans la syntaxe suivante.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

L'authentification à l'aide d'un rôle IAM est présentée dans l'exemple suivant.

```
COPY customer
```

```
FROM 's3://amzn-s3-demo-bucket/mydata'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Pour plus d'informations sur d'autres options d'autorisation, consultez Paramètres d'autorisation

Si vous voulez valider vos données sans charger le tableau, utilisez l'option NOLOAD avec la commande COPY.

L'exemple suivant montre les toutes premières lignes des données, délimitées par une barre verticale, d'un fichier nommé `venue.txt`.

```
1|Toyota Park|Bridgeview|IL|0
2|Columbus Crew Stadium|Columbus|OH|0
3|RFK Stadium|Washington|DC|0
```

Avant le chargement du fichier sur Amazon S3, fractionnez le fichier en plusieurs fichiers, afin que la commande COPY puisse le charger à l'aide du traitement parallèle. Le nombre de fichiers doit être un multiple du nombre de tranches de votre cluster. Fractionnez vos fichiers de données de chargement de telle sorte que les fichiers soient à peu près de taille égale, entre 1 Mo et 1 Go après compression. Pour plus d'informations, consultez Chargement de données à partir de fichiers compressés et non compressés.

Par exemple, le fichier `venue.txt` peut être fractionné en quatre fichiers, comme suit :

```
venue.txt.1
venue.txt.2
venue.txt.3
venue.txt.4
```

La commande COPY suivante charge la table VENUE à l'aide des données délimitées par une barre verticale des fichiers de données avec le préfixe 'venue' dans le compartiment Amazon S3 `amzn-s3-demo-bucket`.

> **Note**
>
> Le compartiment Amazon S3 `amzn-s3-demo-bucket` des exemples suivants n'existe pas. Pour des exemples de commandes COPY qui utilisent les données réelles d'un compartiment Amazon S3 existant, consultez Chargement des exemples de données.

```
COPY venue FROM 's3://amzn-s3-demo-bucket/venue'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
DELIMITER '|';
```

Si aucun objet Amazon S3 avec le préfixe de clé 'venue' n'existe, le chargement échoue.

Rubriques

- Utilisation d'un manifeste pour spécifier les fichiers de données
- Chargement des fichiers de données compressés à partir d'Amazon S3
- Chargement de données de largeur fixe à partir d'Amazon S3
- Chargement de données multioctets à partir d'Amazon S3
- Chargement de fichiers de données chiffrés à partir d'Amazon S3

Utilisation d'un manifeste pour spécifier les fichiers de données

Vous pouvez utiliser un manifeste pour faire en sorte que la commande COPY charge tous les fichiers nécessaires, et seulement ceux-là, lors d'un chargement de données. Vous pouvez utiliser un manifeste pour charger les fichiers de différents compartiments ou les fichiers qui ne partagent pas le même préfixe. Au lieu de fournir un chemin d'objet pour la commande COPY, vous fournissez le nom d'un fichier texte au format JSON qui répertorie explicitement les fichiers à charger. L'URL du manifeste doit spécifier le nom de compartiment et le chemin d'objet complet du fichier, pas simplement un préfixe.

Pour plus d'informations sur les fichiers manifestes, consultez l'exemple de COPY Utilisation d'un manifeste pour spécifier des fichiers de données.

L'exemple suivant illustre le format JSON pour charger les fichiers de différents compartiments et avec des noms de fichiers commençant par des horodatages.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket1/2013-10-04-custdata", "mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket1/2013-10-05-custdata", "mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket2/2013-10-04-custdata", "mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket2/2013-10-05-custdata", "mandatory":true}
  ]
}
```

L'indicateur facultatif `mandatory` spécifie si la commande COPY doit renvoyer une erreur si le fichier est introuvable. La valeur par défaut de `mandatory` est `false`. Quels que soient les paramètres obligatoires, la commande COPY s'arrête si aucun fichier n'est trouvé.

L'exemple suivant exécute la commande COPY avec le manifeste de l'exemple précédent, qui s'appelle `cust.manifest`.

```
COPY customer
FROM 's3://amzn-s3-demo-bucket/cust.manifest'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
MANIFEST;
```

Utilisation d'un manifeste créé par une commande UNLOAD

Il se peut qu'un manifeste créé par une opération UNLOAD utilisant le paramètre MANIFEST contienne des clés non nécessaires à l'opération COPY. Par exemple, le manifeste `UNLOAD` suivant comprend une clé `meta` qui est requise pour une table externe Amazon Redshift Spectrum et pour charger des fichiers de données au format `ORC` ou `Parquet`. La clé `meta` contient une clé `content_length` avec une valeur correspondant à la taille réelle du fichier en octets. L'opération COPY nécessite uniquement la clé `url` et une clé `mandatory` facultative.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/unload/manifest_0000_part_00", "meta":
 { "content_length": 5956875 }},
    {"url":"s3://amzn-s3-demo-bucket/unload/unload/manifest_0001_part_00", "meta":
 { "content_length": 5997091 }}
 ]
}
```

Pour plus d'informations sur les fichiers manifeste, consultez Example: COPY from Amazon S3 using a manifest.

Chargement des fichiers de données compressés à partir d'Amazon S3

Pour charger des fichiers de données compressés à l'aide de gzip, lzop ou bzip2, incluez l'option correspondante : GZIP, LZOP ou. BZIP2

Par exemple, la commande suivante charge des fichiers compressés à l'aide de lzop.

```
COPY customer FROM 's3://amzn-s3-demo-bucket/customer.lzo'
```

```
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
DELIMITER '|' LZOP;
```

> **Note**
>
> Si vous compressez un fichier de données avec la compression lzop et que vous utilisez l'option --filter, la commande COPY ne la prend pas en charge.

Chargement de données de largeur fixe à partir d'Amazon S3

Les fichiers de données de largeur fixe ont des longueurs uniformes pour chaque colonne de données. Chaque champ d'un fichier de données de largeur fixe possède exactement les mêmes longueur et position. Pour les données de type character (CHAR et VARCHAR) d'un fichier de données de largeur fixe, vous devez inclure les espaces de début ou de fin comme espaces réservés afin que la largeur demeure uniforme. Pour les entiers, vous devez utiliser des zéros comme espaces réservés. Un fichier de données de largeur fixe n'a pas de délimiteur pour séparer les colonnes.

Pour charger un fichier de données de largeur fixe dans une table existante, utilisez le paramètre FIXEDWIDTH de la commande COPY. Vos spécifications de table doivent correspondre à la valeur de fixedwidth_spec pour que les données se chargent correctement.

Pour charger les données de largeur fixe depuis un fichier vers une table, émettez la commande suivante :

```
COPY table_name FROM 's3://amzn-s3-demo-bucket/prefix'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
FIXEDWIDTH 'fixedwidth_spec';
```

Le paramètre fixedwidth_spec est une chaîne qui contient un identificateur pour chaque colonne et la largeur de chaque colonne, avec le caractère deux-points comme séparateur. Les paires **`column:width`** sont délimitées par des virgules. L'identificateur peut être l'un de vos choix : chiffres, lettres ou une combinaison des deux. L'identificateur n'ayant aucune relation avec la table elle-même, les spécifications doivent contenir les colonnes dans le même ordre que la table.

Les deux exemples suivants présentent les mêmes spécifications, avec la première utilisant des identificateurs de type numérique (numeric) et la seconde des identificateurs de type chaîne (string) :

```
'0:3,1:25,2:12,3:2,4:6'
```

```
'venueid:3,venuename:25,venuecity:12,venuestate:2,venueseats:6'
```

L'exemple suivant montre un exemple de données de largeur fixe qui pourraient être chargées dans la table VENUE selon les spécifications précédentes :

```
1  Toyota Park               Bridgeview  IL0
2  Columbus Crew Stadium     Columbus    OH0
3  RFK Stadium               Washington  DC0
4  CommunityAmerica Ballpark Kansas City KS0
5  Gillette Stadium          Foxborough  MA68756
```

La commande COPY suivante charge cet ensemble de données dans la table VENUE :

```
COPY venue
FROM 's3://amzn-s3-demo-bucket/data/venue_fw.txt'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
FIXEDWIDTH 'venueid:3,venuename:25,venuecity:12,venuestate:2,venueseats:6';
```

Chargement de données multioctets à partir d'Amazon S3

Si vos données incluent des caractères non-ASCII codés sur plusieurs octets (par exemple, les caractères chinois ou cyrilliques), vous devez charger les données dans des colonnes VARCHAR. Le type de données VARCHAR prend en charge les caractères UTF-8 codés sur quatre octets, mais le type de données CHAR n'accepte que les caractères ASCII codés sur un octet. Vous ne pouvez pas charger de caractères codés sur cinq octets ou plus dans des tables Amazon Redshift. Pour plus d'informations sur CHAR et VARCHAR, consultez Types de données.

Pour vérifier l'encodage utilisé par un fichier d'entrée, choisissez la commande Linux *`file`* :

```
$ file ordersdata.txt
ordersdata.txt: ASCII English text
$ file uni_ordersdata.dat
uni_ordersdata.dat: UTF-8 Unicode text
```

Chargement de fichiers de données chiffrés à partir d'Amazon S3

Vous pouvez utiliser la commande COPY pour charger les fichiers de données qui ont été chargées sur Amazon S3 à l'aide d'un chiffrement côté serveur, d'un chiffrement côté client ou des deux.

La commande COPY prend en charge les types de chiffrement Amazon S3 suivants :

- Chiffrement côté serveur avec des clés gérées par Amazon S3 (SSE-S3)
- Chiffrement côté serveur avec AWS KMS keys (SSE-KMS)
- Chiffrement côté client via une clé racine symétrique côté client

La commande COPY ne prend pas en charge les types de chiffrement Amazon S3 suivants :

- Chiffrement côté serveur avec clés fournies par le client (SSE-C)
- Chiffrement côté client à l'aide d'un AWS KMS key
- Chiffrement côté client via une clé racine asymétrique fournie par le client

Pour plus d'informations sur le chiffrement Amazon S3, consultez Protection des données à l'aide d'un chiffrement côté serveur et Protection des données via le chiffrement côté client dans le Guide de l'utilisateur Amazon Simple Storage Service.

La commande UNLOAD chiffre automatiquement les fichiers à l'aide de SSE-S3. Vous pouvez également effectuer un déchargement avec un chiffrement côté client ou SSE-KMS avec une clé symétrique gérée par le client. Pour de plus amples informations, consultez Déchargement de fichiers de données chiffrés.

La commande COPY reconnaît et charge automatiquement les fichiers chiffrés avec SSE-S3 et SSE-KMS. Vous pouvez charger des fichiers chiffrés à l'aide d'une clé racine symétrique côté client en spécifiant l'option ENCRYPTED et en fournissant la valeur de la clé. Pour plus d'informations, consultez Chargement de données chiffrées sur Amazon S3.

Pour charger des fichiers de données chiffrés côté client, indiquez la valeur de la clé racine à l'aide du paramètre MASTER_SYMMETRIC_KEY et ajoutez l'option ENCRYPTED.

```
COPY customer FROM 's3://amzn-s3-demo-bucket/encrypted/customer'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
MASTER_SYMMETRIC_KEY '<root_key>'
ENCRYPTED
DELIMITER '|';
```

Pour charger des fichiers de données chiffrés compressés au format gzip, lzop ou bzip2, incluez l'option GZIP, LZOP ou ainsi que la valeur de la clé racine et l' BZIP2 option ENCRYPTED.

```
COPY customer FROM 's3://amzn-s3-demo-bucket/encrypted/customer'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
MASTER_SYMMETRIC_KEY '<root_key>'
ENCRYPTED
DELIMITER '|'
GZIP;
```

Chargement de données à partir d'Amazon EMR

Vous pouvez utiliser la commande COPY pour charger des données en parallèle à partir d'un cluster Amazon EMR configuré pour écrire les fichiers texte dans le système de fichiers distribué Hadoop (HDFS) du cluster sous la forme de fichiers à largeur fixe, de fichiers délimités par des caractères, de fichiers CSV ou de fichiers au format JSON.

Processus de chargement des données à partir d'Amazon EMR

Cette section vous guide à travers le processus de chargement des données à partir d'un cluster Amazon EMR. Les sections suivantes fournissent les informations détaillées dont vous avez besoin pour effectuer chaque étape.

- Étape 1 : Configurer des autorisations IAM

  Les utilisateurs qui créent le cluster Amazon EMR et exécutent la commande Amazon Redshift COPY doivent avoir les autorisations nécessaires.
- Étape 2 : Créer un cluster Amazon EMR

  Configurez le cluster pour produire les fichiers texte sur le système de fichiers DFS Hadoop (HDFS). Vous aurez besoin de l'ID du cluster Amazon EMR et du DNS public principal du cluster (le point de terminaison de l' EC2 instance Amazon qui héberge le cluster).
- Étape 3 : Récupérer la clé publique de cluster Amazon Redshift et les adresses IP de nœud de cluster

  La clé publique permet aux nœuds de cluster Amazon Redshift d'établir des connexions SSH aux hôtes. Vous allez utiliser l'adresse IP de chaque nœud de cluster pour configurer les groupes de sécurité hôte et permettre l'accès à partir de votre cluster Amazon Redshift à l'aide de ces adresses IP.
- Étape 4 : ajouter la clé publique du cluster Amazon Redshift au fichier de clés autorisées de EC2 chaque hôte Amazon

Vous ajoutez la clé publique de cluster Amazon Redshift au fichier des clés autorisées de l'hôte de telle sorte que l'hôte reconnaisse le cluster Amazon Redshift et accepte la connexion SSH.

- Étape 5 : Configurer les hôtes pour accepter toutes les adresses IP du cluster Amazon Redshift

  Modifiez les groupes de sécurité de l'instance Amazon EMR pour ajouter des règles de trafic entrant et accepter les adresses IP Amazon Redshift.

- Étape 6 : Exécuter la commande COPY pour charger les données

  Depuis une base de données Amazon Redshift, exécutez la commande COPY pour charger les données dans une table Amazon Redshift.

Étape 1 : Configurer des autorisations IAM

Les utilisateurs qui créent le cluster Amazon EMR et exécutent la commande Amazon Redshift COPY doivent avoir les autorisations nécessaires.

Pour configurer les autorisations IAM

1. Ajoutez les autorisations suivantes pour l'utilisateur qui créera le cluster Amazon EMR.

   ```
   ec2:DescribeSecurityGroups
   ec2:RevokeSecurityGroupIngress
   ec2:AuthorizeSecurityGroupIngress
   redshift:DescribeClusters
   ```

2. Ajoutez l'autorisation suivante pour le rôle IAM ou l'utilisateur qui exécutera la commande COPY.

   ```
   elasticmapreduce:ListInstances
   ```

3. Ajoutez l'autorisation suivante au rôle IAM du cluster Amazon EMR.

   ```
   redshift:DescribeClusters
   ```

Étape 2 : Créer un cluster Amazon EMR

La commande COPY charge les données à partir des fichiers du système de fichiers distribué Hadoop (HDFS) Amazon EMR. Lorsque vous créez le cluster Amazon EMR, configurez le cluster pour générer les fichiers de données sur le système de fichiers distribué Hadoop (HDFS) du cluster.

Pour créer un cluster Amazon EMR

1. Créez un cluster Amazon EMR dans la même AWS région que le cluster Amazon Redshift.

   Si le cluster Amazon Redshift est un VPC, le cluster Amazon EMR doit être dans le même groupe VPC. Si le cluster Amazon Redshift utilise le mode EC2 -Classic (c'est-à-dire qu'il ne se trouve pas dans un VPC), le cluster Amazon EMR doit également utiliser le mode -Classic. EC2 Pour plus d'informations, consultez Gestion des clusters dans un cloud privé virtuel (VPC) dans le Guide de gestion Amazon Redshift.

2. Configurez le cluster pour générer les fichiers de données sur le système de fichiers distribué Hadoop (HDFS) du cluster. Les noms de fichiers HDFS ne doivent pas comporter d'astérisque (*) ou de point d'interrogation (?).

   > **Important**
   >
   > Les noms de fichiers ne doivent pas comporter d'astérisque (*) ou de point d'interrogation (?).

3. Spécifiez No (Non) pour l'option Auto-terminate (Arrêt automatique) de la configuration du cluster Amazon EMR de telle sorte que le cluster demeure disponible pendant que la commande COPY s'exécute.

   > **Important**
   >
   > Si l'un des fichiers de données est modifié ou supprimé avant la fin de la commande COPY, vous pouvez avoir des résultats inattendus ou l'opération COPY peut échouer.

4. Notez l'ID du cluster et le DNS public principal (le point de terminaison de l' EC2instance Amazon qui héberge le cluster). Vous allez utiliser ces informations dans les étapes ultérieures.

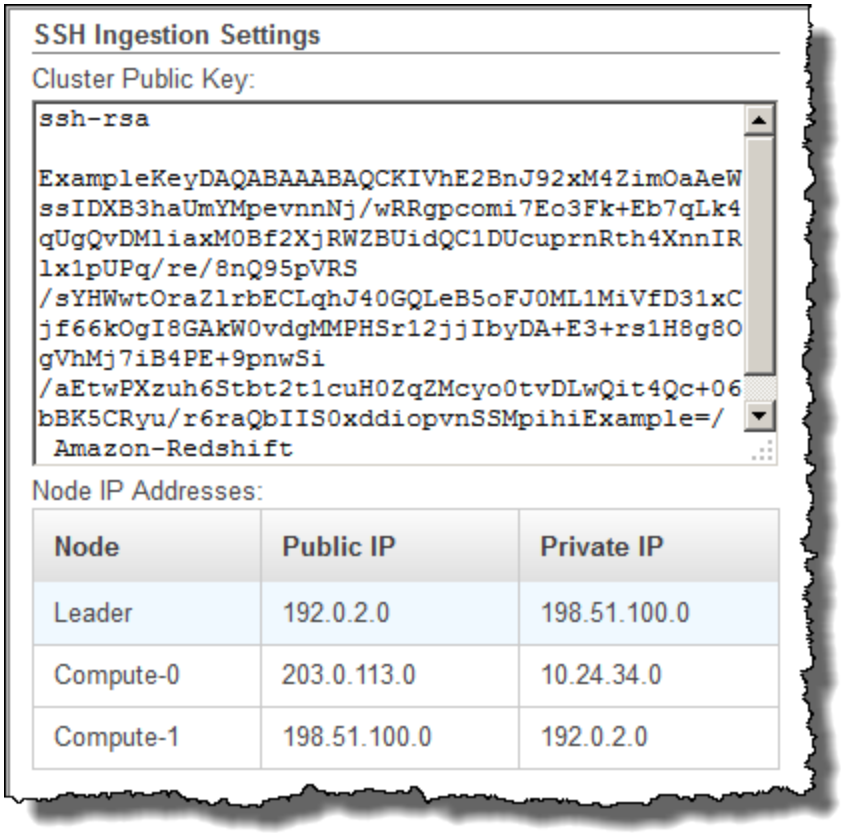
Étape 3 : Récupérer la clé publique de cluster Amazon Redshift et les adresses IP de nœud de cluster

Vous allez utiliser l'adresse IP de chaque nœud de cluster pour configurer les groupes de sécurité hôte et permettre l'accès à partir de votre cluster Amazon Redshift à l'aide de ces adresses IP.

Pour récupérer la clé publique de cluster et les adresses IP de nœud de cluster Amazon Redshift pour votre cluster à l'aide de la console

1. Accédez à la console de gestion Amazon Redshift.
2. Choisissez le lien Clusters dans le volet de navigation.
3. Sélectionnez votre cluster dans la liste.
4. Recherchez le groupe Paramètres d'ingestion SSH.

   Notez la Clé publique du cluster et les Adresses IP du nœud. Vous allez les utiliser dans les étapes ultérieures.



   Vous utiliserez les adresses IP privées de l'étape 3 pour configurer l' EC2 hôte Amazon afin qu'il accepte la connexion depuis Amazon Redshift.

Pour récupérer la clé publique de cluster et les adresses IP de nœud de cluster pour votre cluster à l'aide de la CLI Amazon Redshift, exécutez la commande describe-clusters. Par exemple :

```
aws redshift describe-clusters --cluster-identifier <cluster-identifier>
```

La réponse inclura une ClusterPublicKey valeur et la liste des adresses IP privées et publiques, similaires à ce qui suit :

```
{
    "Clusters": [
        {
            "VpcSecurityGroups": [],
            "ClusterStatus": "available",
            "ClusterNodes": [
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "LEADER",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-0",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-1",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                }
            ],
            "AutomatedSnapshotRetentionPeriod": 1,
            "PreferredMaintenanceWindow": "wed:05:30-wed:06:00",
            "AvailabilityZone": "us-east-1a",
            "NodeType": "dc2.large",
            "ClusterPublicKey": "ssh-rsa AAAABexamplepublickey...Y3TAl Amazon-
Redshift",
             ...
             ...
}
```

Pour extraire la clé publique de cluster et les adresses IP de nœud de cluster de votre cluster à l'aide de l'API Amazon Redshift, utilisez l'action `DescribeClusters`. Pour plus d'informations, consultez la section describe-clusters dans le guide de l'interface de ligne de commande Amazon Redshift DescribeClustersou dans le guide de l'API Amazon Redshift.

Étape 4 : ajouter la clé publique du cluster Amazon Redshift au fichier de clés autorisées de EC2 chaque hôte Amazon

Vous ajoutez la clé publique de cluster à chaque fichier de clés autorisées de l'hôte pour tous les nœuds de cluster Amazon EMR de telle sorte que les hôtes reconnaissent Amazon Redshift et acceptent la connexion SSH.

Pour ajouter la clé publique de cluster Amazon Redshift au fichier de clés autorisées de l'hôte

1. Accédez à l'hôte à l'aide d'une connexion SSH.

   Pour plus d'informations sur la connexion à une instance via SSH, consultez Connect to Your Instance dans le guide de l' EC2 utilisateur Amazon.
2. Copiez la clé publique Amazon Redshift à partir de la console ou du texte de réponse de la CLI.
3. Copiez et collez le contenu de la clé publique dans le fichier `/home/<ssh_username>/.ssh/authorized_keys` de l'hôte. Incluez la chaîne complète, y compris le préfixe « `ssh-rsa` » et le suffixe « `Amazon-Redshift` ». Par exemple :

   ```
   ssh-rsa AAAACTP3isxgGzVWoIWpbVvRCOzYdVifMrh… uA70BnMHCaMiRdmvsDOedZDOedZ Amazon-
   Redshift
   ```

Étape 5 : Configurer les hôtes pour accepter toutes les adresses IP du cluster Amazon Redshift

Pour autoriser le trafic entrant des instances de l'hôte, modifiez le groupe de sécurité et ajoutez une règle de trafic entrant pour chacun nœud de cluster Amazon Redshift. Pour Type, sélectionnez SSH avec le protocole TCP sur le port 22. Pour Source, saisissez les adresses IP privées du nœud de cluster Amazon Redshift que vous avez récupérées dans Étape 3 : Récupérer la clé publique de cluster Amazon Redshift et les adresses IP de nœud de cluster. Pour plus d'informations sur l'ajout de règles à un groupe EC2 de sécurité Amazon, consultez Autoriser le trafic entrant pour vos instances dans le guide de EC2 l'utilisateur Amazon.

Étape 6 : Exécuter la commande COPY pour charger les données

Exécutez une commande COPY pour vous connecter au cluster Amazon EMR et charger les données dans une table Amazon Redshift. Le cluster Amazon EMR doit continuer à s'exécuter jusqu'à la fin de la commande COPY. Par exemple, ne configurez pas le cluster pour qu'il se termine automatiquement.

> **Important**
>
> Si l'un des fichiers de données est modifié ou supprimé avant la fin de la commande COPY, vous pouvez avoir des résultats inattendus ou l'opération COPY peut échouer.

Dans la commande COPY, spécifiez l'ID de cluster Amazon EMR, ainsi que le nom et le chemin d'accès du fichier HDFS.

```
COPY sales
FROM 'emr://myemrclusterid/myoutput/part*' CREDENTIALS
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Vous pouvez utiliser l'astérisque comme caractère générique (`*`) et le point d'interrogation (`?`) dans le cadre de l'argument de nom de fichier. Par exemple, `part*` charge les fichiers `part-0000`, `part-0001`, et ainsi de suite. Si vous spécifiez uniquement un nom de dossier, COPY tente de charger tous les fichiers dans le dossier.

> **Important**
>
> Si vous utilisez des caractères génériques ou utilisez uniquement le nom de dossier, vérifiez qu'aucun fichier indésirable n'est chargées, sinon la commande COPY échoue. Par exemple, certains processus peuvent écrire un fichier journal sur le dossier de sortie.

Chargement des données à partir des hôtes distants

Vous pouvez utiliser la commande COPY pour charger des données en parallèle depuis un ou plusieurs hôtes distants, tels que des EC2 instances Amazon ou d'autres ordinateurs. COPY se connecte aux hôtes distants à l'aide de SSH et exécute les commandes sur les hôtes distants pour générer la sortie texte.

L'hôte distant peut être une instance Amazon EC2 Linux ou un autre ordinateur Unix ou Linux configuré pour accepter les connexions SSH. Ce guide part du principe que votre hôte distant est une EC2 instance Amazon. Lorsque la procédure est différente pour un autre ordinateur, le guide signale la différence.

Amazon Redshift peut se connecter à plusieurs hôtes et ouvrir plusieurs connexions SSH à chaque hôte. Amazon Redshift envoie une commande unique via chaque connexion pour générer la sortie texte sur la sortie standard de l'hôte, qu'Amazon Redshift lit ensuite comme un fichier texte.

Avant de commencer

Avant de commencer, vous devez avoir les éléments suivants en place :

- Une ou plusieurs machines hôtes, telles que des EC2 instances Amazon, auxquelles vous pouvez vous connecter via SSH.
- Sources de données sur les hôtes.

  Vous devez fournir les commandes que le cluster Amazon Redshift exécutera sur les hôtes pour générer la sortie texte. Une fois que le cluster s'est connecté à un hôte, la commande COPY exécute les commandes, lit le texte depuis la sortie standard des hôtes et charge les données en parallèle dans une table Amazon Redshift. La sortie texte doit être sous une forme que la commande COPY peut assimiler. Pour plus d'informations, consultez Préparation de vos données d'entrée
- Accédez aux hôtes à partir de votre ordinateur.

  Pour une EC2 instance Amazon, vous utiliserez une connexion SSH pour accéder à l'hôte. Vous devez accéder à l'hôte pour ajouter la clé publique du cluster Amazon Redshift au fichier de clés autorisées de l'hôte.
- Un cluster Amazon Redshift en cours d'exécution.

  Pour plus d'informations sur le lancement d'un cluster, consultez Guide de démarrage d'Amazon Redshift.

Processus de chargement de données

Cette section vous guide à travers le processus de chargement de données à partir d'hôtes distants. Les sections suivantes fournissent les informations détaillées dont vous avez besoin pour effectuer chaque étape.

- Étape 1 : Récupérer la clé publique de cluster et les adresses IP de nœud de cluster

  La clé publique permet aux nœuds de cluster Amazon Redshift d'établir des connexions SSH aux hôtes distants. Vous allez utiliser l'adresse IP de chaque nœud de cluster pour configurer

les groupes de sécurité hôte ou le pare-feu, et permettre l'accès à partir de votre cluster Amazon Redshift à l'aide de ces adresses IP.

- Étape 2 : Ajouter la clé publique de cluster Amazon Redshift au fichier de clés autorisées de l'hôte

  Vous ajoutez la clé publique de cluster Amazon Redshift au fichier des clés autorisées de l'hôte de telle sorte que l'hôte reconnaisse le cluster Amazon Redshift et accepte la connexion SSH.

- Étape 3 : Configurer l'hôte pour accepter toutes les adresses IP du cluster Amazon Redshift

  Pour Amazon EC2, modifiez les groupes de sécurité de l'instance pour ajouter des règles de saisie afin d'accepter les adresses IP Amazon Redshift. Pour les autres hôtes, modifiez le pare-feu de telle sorte que vos nœuds Amazon Redshift puissent établir des connexions SSH à l'hôte distant.

- Étape 4 : Obtenir la clé publique de l'hôte

  Vous pouvez spécifier le cas échéant qu'Amazon Redshift doit utiliser la clé publique pour identifier l'hôte. Vous devez trouver la clé publique et copier le texte dans votre fichier manifeste.

- Étape 5 : Créer un fichier manifeste

  Le manifeste est un fichier texte au format JSON avec les détails dont Amazon Redshift a besoin pour se connecter aux hôtes et récupérer les données.

- Étape 6 : charger le fichier manifeste sur un compartiment Amazon S3

  Amazon Redshift lit le manifeste et utilise ces informations pour se connecter à l'hôte distant. Si le compartiment Amazon S3 ne réside pas dans la même région que votre cluster Amazon Redshift, vous devez utiliser l'option REGION pour spécifier la région dans laquelle les données se trouvent.

- Étape 7 : Exécuter la commande COPY pour charger les données

  Depuis une base de données Amazon Redshift, exécutez la commande COPY pour charger les données dans une table Amazon Redshift.

Étape 1 : Récupérer la clé publique de cluster et les adresses IP de nœud de cluster

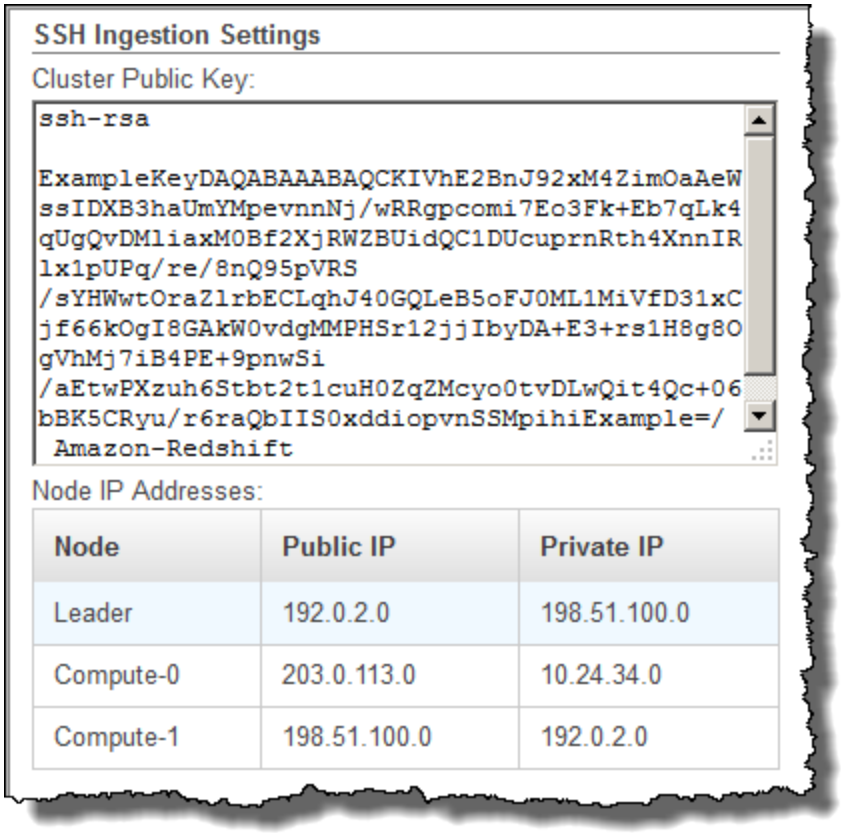
Vous allez utiliser l'adresse IP de chaque nœud de cluster pour configurer les groupes de sécurité hôte et permettre l'accès à partir de votre cluster Amazon Redshift à l'aide de ces adresses IP.

Pour récupérer la clé publique de cluster et les adresses IP de nœud de cluster pour votre cluster à l'aide de la console

1. Accédez à la console de gestion Amazon Redshift.

2. Choisissez le lien Clusters dans le volet de navigation.
3. Sélectionnez votre cluster dans la liste.
4. Recherchez le groupe Paramètres d'ingestion SSH.

   Notez la Clé publique du cluster et les Adresses IP du nœud. Vous allez les utiliser dans les étapes ultérieures.



SSH Ingestion Settings

Cluster Public Key:

```
ssh-rsa

ExampleKeyDAQABAAABAQCKIVhE2BnJ92xM4ZimOaAeW
ssIDXB3haUmYMpevnnNj/wRRgpcomi7Eo3Fk+Eb7qLk4
qUgQvDMliaxM0Bf2XjRWZBUidQC1DUcuprnRth4XnnIR
lx1pUPq/re/8nQ95pVRS
/sYHWwtOraZlrbECLqhJ40GQLeB5oFJ0ML1MiVfD31xC
jf66kOgI8GAkW0vdgMMPHSr12jjIbyDA+E3+rs1H8g8O
gVhMj7iB4PE+9pnwSi
/aEtwPXzuh6Stbt2t1cuH0ZqZMcyo0tvDLwQit4Qc+06
bBK5CRyu/r6raQbIIS0xddiopvnSSMpihiExample=/
 Amazon-Redshift
```

Node IP Addresses:

Node	Public IP	Private IP
Leader	192.0.2.0	198.51.100.0
Compute-0	203.0.113.0	10.24.34.0
Compute-1	198.51.100.0	192.0.2.0

   Vous allez utiliser les adresses IP privées de l'étape 3 pour configurer l'hôte et accepter la connexion à partir d'Amazon Redshift. Selon le type d'hôte auquel vous vous connectez à et qu'il figure ou pas dans un VPC, vous allez utiliser les adresses IP publiques ou les adresses IP privées.

Pour récupérer la clé publique de cluster et les adresses IP de nœud de cluster pour votre cluster à l'aide de la CLI Amazon Redshift, exécutez la commande describe-clusters.

Par exemple :

```
aws redshift describe-clusters --cluster-identifier <cluster-identifier>
```

La réponse inclura ClusterPublicKey la liste des adresses IP privées et publiques, comme suit :

```
{
    "Clusters": [
        {
            "VpcSecurityGroups": [],
            "ClusterStatus": "available",
            "ClusterNodes": [
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "LEADER",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-0",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                },
                {
                    "PrivateIPAddress": "10.nnn.nnn.nnn",
                    "NodeRole": "COMPUTE-1",
                    "PublicIPAddress": "10.nnn.nnn.nnn"
                }
            ],
            "AutomatedSnapshotRetentionPeriod": 1,
            "PreferredMaintenanceWindow": "wed:05:30-wed:06:00",
            "AvailabilityZone": "us-east-1a",
            "NodeType": "dc2.large",
            "ClusterPublicKey": "ssh-rsa AAAABexamplepublickey...Y3TAl Amazon-
Redshift",
             ...
             ...
}
```

Pour récupérer la clé publique du cluster et les adresses IP des nœuds de cluster de votre cluster à l'aide de l'API Amazon Redshift, utilisez l' DescribeClusters action. Pour plus d'informations, consultez la section describe-clusters dans le guide de l'interface de ligne de commande Amazon Redshift DescribeClustersou dans le guide de l'API Amazon Redshift.

Étape 2 : Ajouter la clé publique de cluster Amazon Redshift au fichier de clés autorisées de l'hôte

Vous ajoutez la clé publique de cluster à chaque fichier de clés autorisées de l'hôte de telle sorte que l'hôte reconnaisse Amazon Redshift et accepte la connexion SSH.

Pour ajouter la clé publique de cluster Amazon Redshift au fichier de clés autorisées de l'hôte

1. Accédez à l'hôte à l'aide d'une connexion SSH.

   Pour plus d'informations sur la connexion à une instance via SSH, consultez Connect to Your Instance dans le guide de l' EC2 utilisateur Amazon.
2. Copiez la clé publique Amazon Redshift à partir de la console ou du texte de réponse de la CLI.
3. Copiez et collez le contenu de la clé publique dans le fichier `/home/<ssh_username>/.ssh/authorized_keys` de l'hôte distant. `<ssh_username>` doit correspondre à la valeur du champ « username » du fichier manifeste. Incluez la chaîne complète, y compris le préfixe « `ssh-rsa` » et le suffixe « `Amazon-Redshift` ». Par exemple :

   ```
   ssh-rsa AAAACTP3isxgGzVWoIWpbVvRCOzYdVifMrh… uA70BnMHCaMiRdmvsDOedZDOedZ Amazon-
   Redshift
   ```

Étape 3 : Configurer l'hôte pour accepter toutes les adresses IP du cluster Amazon Redshift

Si vous travaillez avec une EC2 instance Amazon ou un cluster Amazon EMR, ajoutez des règles de trafic entrant au groupe de sécurité de l'hôte pour autoriser le trafic provenant de chaque nœud de cluster Amazon Redshift. Pour Type, sélectionnez SSH avec le protocole TCP sur le port 22. Pour Source, saisissez les adresses IP du nœud de cluster Amazon Redshift que vous avez récupérées dans Étape 1 : Récupérer la clé publique de cluster et les adresses IP de nœud de cluster. Pour plus d'informations sur l'ajout de règles à un groupe EC2 de sécurité Amazon, consultez Autoriser le trafic entrant pour vos instances dans le guide de EC2 l'utilisateur Amazon.

Utilisez les adresses IP privées quand :

- Vous avez un cluster Amazon Redshift qui ne se trouve pas dans un Virtual Private Cloud (VPC) et une instance EC2 Amazon -Classic, tous deux situés dans la même région. AWS

- Vous avez un cluster Amazon Redshift situé dans un VPC et une instance Amazon EC2 -VPC, tous deux situés dans la même région AWS et dans le même VPC.

Sinon, utilisez les adresses IP publiques.

Pour plus d'informations sur l'utilisation d'Amazon Redshift dans un VPC, consultez Gestion des clusters dans un cloud privé virtuel (VPC) dans le Guide de gestion Amazon Redshift.

Étape 4 : Obtenir la clé publique de l'hôte

Vous pouvez fournir le cas échéant la clé publique dans le fichier manifeste de telle sorte qu'Amazon Redshift puisse identifier l'hôte. La commande COPY ne nécessite pas la clé publique de l'hôte mais, pour des raisons de sécurité, nous vous recommandons vivement d'utiliser une clé publique pour empêcher les attaques man-in-the-middle « ».

Vous pouvez trouver la clé publique de l'hôte à l'emplacement suivant, où `<ssh_host_rsa_key_name>` correspond au nom unique de la clé publique de l'hôte :

```
:  /etc/ssh/<ssh_host_rsa_key_name>.pub
```

> **Note**
>
> Amazon Redshift prend uniquement en charge les clés RSA. Nous ne prenons pas en charge les clés DSA.

Lorsque vous créez votre fichier manifeste à l'étape 5, vous collez le texte de la clé publique dans le champ « Public Key » de l'entrée du fichier manifeste.

Étape 5 : Créer un fichier manifeste

La commande COPY peut se connecter à plusieurs hôtes à l'aide de SSH et créer plusieurs connexions SSH à chaque hôte. COPY exécute une commande via chaque connexion hôte, puis charge la sortie à partir des commandes en parallèle de la table. Le fichier manifeste est un fichier texte au format JSON qu'Amazon Redshift utilise pour se connecter à l'hôte. Le fichier manifeste spécifie les points de terminaison hôte SSH et les commandes qui sont exécutées sur les hôtes pour renvoyer les données à Amazon Redshift. Le cas échéant, vous pouvez inclure la clé publique de l'hôte, le nom d'utilisateur de connexion et un indicateur obligatoire pour chaque entrée.

Créez le fichier manifeste sur votre ordinateur local. Dans une étape ultérieure, vous téléchargez le fichier vers Amazon S3.

Le fichier manifeste est au format suivant :

```
{
   "entries": [
     {"endpoint":"<ssh_endpoint_or_IP>",
       "command": "<remote_command>",
       "mandatory":true,
       "publickey": "<public_key>",
       "username": "<host_user_name>"},
     {"endpoint":"<ssh_endpoint_or_IP>",
       "command": "<remote_command>",
       "mandatory":true,
       "publickey": "<public_key>",
       "username": "host_user_name"}
    ]
}
```

Le fichier manifeste contient une construction « entries » pour chaque connexion SSH. Chaque entrée représente une seule connexion SSH. Vous pouvez avoir plusieurs connexions à un seul hôte ou plusieurs connexions à plusieurs hôtes. Les guillemets doubles sont obligatoires comme illustré, aussi bien pour les noms de champ que pour les valeurs. La seule valeur qui n'a pas besoin de guillemets doubles est la valeur booléenne **`true`** ou **`false`** pour le champ obligatoire.

La liste suivante décrit les champs dans le fichier manifeste.

point de terminaison

Adresse URL ou adresse IP de l'hôte. Par exemple, « `ec2-111-222-333.compute-1.amazonaws.com` » ou « `22.33.44.56` »

command

La commande qui sera exécutée par l'hôte pour générer une sortie texte ou binaire (lzop, gzip ou bzip2). La commande peut être n'importe quelle commande que l'utilisateur « host_user_name » est autorisé à exécuter. La commande peut être aussi simple que l'impression d'un fichier ou peut interroger une base de données ou lancer un script. La sortie (fichier texte, fichier binaire gzip, fichier binaire lzop ou fichier binaire bzip2) doit être sous une forme que la commande COPY Amazon Redshift peut intégrer. Pour plus d'informations, consultez Préparation de vos données d'entrée.

publickey

(Facultatif) La clé publique de l'hôte. Si la clé est fournie, Amazon Redshift l'utilise pour identifier l'hôte. Si la clé publique n'est pas fournie, Amazon Redshift n'essaie pas d'identifier l'hôte. Par exemple, si la clé publique de l'hôte distant est `ssh-rsa AbcCbaxxx…xxxDHKJ root@amazon.com`, saisissez le texte suivant dans le champ de clé publique : `AbcCbaxxx…xxxDHKJ`.

mandatory

(Facultatif) Indique si la commande COPY doit échouer en cas d'échec de la connexion. L'argument par défaut est `false`. Si Amazon Redshift n'établit pas avec succès au moins une connexion, la commande COPY échoue.

nom d'utilisateur

(Facultatif) Nom d'utilisateur qui sera utilisé pour vous connecter au système hôte et exécuter la commande à distance. Le nom de connexion d'utilisateur doit être le même que celui de la connexion utilisée pour ajouter la clé publique au fichier de clés autorisées de l'hôte à l'étape 2. Le nom d'utilisateur par défaut est « redshift ».

L'exemple suivant illustre un manifeste complet permettant d'ouvrir quatre connexions vers le même hôte et d'exécuter une commande différente sur chaque connexion :

```
{
  "entries": [
       {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
          "command": "cat loaddata1.txt",
          "mandatory":true,
          "publickey": "ec2publickeyportionoftheec2keypair",
          "username": "ec2-user"},
       {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
          "command": "cat loaddata2.txt",
          "mandatory":true,
          "publickey": "ec2publickeyportionoftheec2keypair",
          "username": "ec2-user"},
       {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
          "command": "cat loaddata3.txt",
          "mandatory":true,
          "publickey": "ec2publickeyportionoftheec2keypair",
          "username": "ec2-user"},
       {"endpoint":"ec2-184-72-204-112.compute-1.amazonaws.com",
```

```
          "command": "cat loaddata4.txt",
          "mandatory":true,
          "publickey": "ec2publickeyportionoftheec2keypair",
          "username": "ec2-user"}
     ]
}
```

Étape 6 : charger le fichier manifeste sur un compartiment Amazon S3

Chargez le fichier manifeste sur un compartiment Amazon S3. Si le compartiment Amazon S3 ne réside pas dans la même AWS région que votre cluster Amazon Redshift, vous devez utiliser l'REGIONoption pour spécifier la AWS région dans laquelle se trouve le manifeste. Pour plus d'informations sur la création d'un compartiment Amazon S3 et le chargement d'un fichier, consultez le Guide de l'utilisateur Amazon Simple Storage Service.

Étape 7 : Exécuter la commande COPY pour charger les données

Exécutez une commande COPY pour vous connecter à l'hôte et charger les données dans une table Amazon Redshift. Dans la commande COPY, spécifiez le chemin d'accès à l'objet Amazon S3 explicite pour le fichier manifeste et incluez l'option SSH. Par exemple,

```
COPY sales
FROM 's3://amzn-s3-demo-bucket/ssh_manifest'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
DELIMITER '|'
SSH;
```

> **Note**
>
> Si vous utilisez la compression automatique, la commande COPY effectue deux lectures des données, ce qui signifie qu'elle exécute la commande distante à deux reprises. La première lecture consiste à fournir un échantillon pour l'analyse de la compression, puis la deuxième lecture charge réellement les données. Si la double exécution de la commande distante cause un problème en raison des effets secondaires potentiels, vous devez désactiver la compression automatique. Pour désactiver la compression automatique, exécutez la commande COPY avec l'option COMPUPDATE définie sur OFF. Pour de plus amples informations, veuillez consulter Chargement des tables avec compression automatique.

Chargement de données à partir d'une table Amazon DynamoDB

Vous pouvez utiliser la commande COPY pour charger une table avec les données d'une seule table Amazon DynamoDB.

> **Important**
>
> La table Amazon DynamoDB qui fournit les données doit être créée dans la AWS même région que votre cluster, sauf si vous utilisez REGION l'option permettant de spécifier la région dans laquelle se AWS trouve la table Amazon DynamoDB.

La commande COPY utilise l'architecture de traitement hautement parallèle (MPP) d'Amazon Redshift pour lire et charger les données en parallèle à partir d'une table Amazon DynamoDB. Vous pouvez profiter au maximum du traitement parallèle en définissant les styles de distribution de vos tables Amazon Redshift. Pour plus d'informations, consultez Distribution des données pour l'optimisation des requêtes.

> **Important**
>
> Lorsque la commande COPY lit des données à partir de la table Amazon DynamoDB, le transfert de données qui en résulte fait partie du débit alloué de la table.

Pour éviter de consommer des quantités excessives de débit alloué en lecture, nous vous recommandons de ne pas charger les données à partir des tables Amazon DynamoDB qui se trouvent dans des environnements de production. Si vous chargez les données à partir des tables de production, nous vous recommandons de définir l'option READRATIO avec une valeur beaucoup plus basse que le pourcentage moyen de débit alloué inutilisé. Un paramètre READRATIO bas contribue à réduire les problèmes de limitation. Pour utiliser la totalité du débit alloué d'une table Amazon DynamoDB, définissez READRATIO avec la valeur 100.

La commande COPY met en correspondance les noms d'attribut des éléments extraits de la table Amazon DynamoDB et les noms de colonne d'une table Amazon Redshift existante en utilisant les règles suivantes :

- Les colonnes de la table Amazon Redshift sont mises en correspondance sans sensibilité à la casse avec les attributs d'élément Amazon DynamoDB. Si un élément de la table DynamoDB

contient plusieurs attributs qui diffèrent uniquement par la casse, comme Prix et PRIX, la commande COPY échoue.

- Les colonnes de la table Amazon Redshift qui ne correspondent pas à un attribut de la table Amazon DynamoDB sont chargées en tant que NULL ou vides, en fonction de la valeur spécifiée par l'option EMPTYASNULL de la commande COPY.
- Les attributs Amazon DynamoDB qui ne correspondent pas à une colonne de la table Amazon Redshift sont ignorés. Les attributs sont lus avant d'être mis en correspondance, et à fortiori les attributs ignorés consomment une partie du débit alloué de la table.
- Seuls les attributs Amazon DynamoDB avec les types de données STRING et NUMBER sont pris en charge. Les types de données Amazon DynamoDB BINARY et SET ne sont pas pris en charge. Si une commande COPY tente de charger un attribut avec un type de données non pris en charge, la commande échoue. Si l'attribut ne correspond pas à une colonne de table Amazon Redshift, COPY n'essaie pas de le charger et aucune erreur n'est déclenchée.

La commande COPY utilise la syntaxe suivante pour charger les données d'une table Amazon DynamoDB :

```
COPY <redshift_tablename> FROM 'dynamodb://<dynamodb_table_name>'
authorization
readratio '<integer>';
```

Les valeurs d'autorisation sont les AWS informations d'identification nécessaires pour accéder à la table Amazon DynamoDB. Si ces informations d'identification correspondent à un utilisateur, cet utilisateur doit avoir la permission d'exécuter les opérations SCAN et DESCRIBE sur la table Amazon DynamoDB en cours de chargement.

Les valeurs d'autorisation fournissent l' AWS autorisation dont votre cluster a besoin pour accéder à la table Amazon DynamoDB. Elle doit comprendre l'autorisation d'exécuter SCAN et DESCRIBE sur la table Amazon DynamoDB qui est en cours de chargement. Pour plus d'informations sur les autorisations requises, consultez Autorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY. La méthode d'authentification recommandée est celle qui consiste à spécifier le paramètre IAM_ROLE et à fournir l'Amazon Resource Name (ARN) d'un rôle IAM avec les autorisations nécessaires. Pour de plus amples informations, veuillez consulter Contrôle d'accès basé sur les rôles.

Pour vous authentifier à l'aide du paramètre IAM_ROLE, *<aws-account-id>* et *<role-name>* comme indiqué dans la syntaxe suivante.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

L'authentification à l'aide d'un rôle IAM est présentée dans l'exemple suivant.

```
COPY favoritemovies
FROM 'dynamodb://ProductCatalog'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Pour plus d'informations sur d'autres options d'autorisation, consultez Paramètres d'autorisation

Si vous voulez valider vos données sans charger le tableau, utilisez l'option NOLOAD avec la commande COPY.

L'exemple suivant charge la table FAVORITEMOVIES avec les données de la table DynamoDB. my-favorite-movies-table L'activité de lecture peut consommer jusqu'à 50 % du débit alloué.

```
COPY favoritemovies FROM 'dynamodb://my-favorite-movies-table'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
READRATIO 50;
```

Pour optimiser le débit, la commande COPY charge les données d'une table Amazon DynamoDB en parallèle sur les nœuds de calcul du cluster.

Débit alloué avec compression automatique

Par défaut, la commande COPY applique la compression automatique chaque fois que vous spécifiez une table cible vide sans encodage de compression. La compression automatique analyse initialement un grand nombre de lignes de la table Amazon DynamoDB. La taille de l'échantillon repose sur la valeur du paramètre COMPROWS. La valeur par défaut est de 100 000 lignes par tranche.

Après le prélèvement, les lignes de l'exemple sont ignorées et la totalité de la table est chargée. Par conséquent, la plupart des lignes sont lues deux fois. Pour plus d'informations sur le fonctionnement de la compression, consultez Chargement des tables avec compression automatique.

> **Important**
>
> Lorsque la commande COPY lit les données de la table Amazon DynamoDB, y compris les lignes utilisées pour les prélèvements, le transfert des données obtenues fait partie du débit alloué de la table.

Chargement de données multioctets à partir d'Amazon DynamoDB

Si vos données incluent des caractères non-ASCII codés sur plusieurs octets (par exemple, les caractères chinois ou cyrilliques), vous devez charger les données dans des colonnes VARCHAR. Le type de données VARCHAR prend en charge les caractères UTF-8 codés sur quatre octets, mais le type de données CHAR n'accepte que les caractères ASCII codés sur un octet. Vous ne pouvez pas charger de caractères codés sur cinq octets ou plus dans des tables Amazon Redshift. Pour plus d'informations sur CHAR et VARCHAR, consultez Types de données.

Vérification que les données ont été chargées correctement

Une fois l'opération de chargement terminée, interrogez la table système STL_LOAD_COMMITS pour vérifier que les fichiers attendus ont été chargés. Exécutez la commande COPY et la vérification de la charge dans la même transaction afin qu'en cas de problème avec la charge, vous puissiez restaurer l'ensemble de la transaction.

La requête suivante renvoie les entrées pour charger les tables dans la base de données TICKIT :

```
SELECT query, trim(filename) AS filename, curtime, status
FROM stl_load_commits
WHERE filename like '%tickit%' order by query;


 query |         filename          |          curtime           | status
-------+---------------------------+----------------------------+--------
 22475 | tickit/allusers_pipe.txt  | 2013-02-08 20:58:23.274186 |      1
 22478 | tickit/venue_pipe.txt     | 2013-02-08 20:58:25.070604 |      1
 22480 | tickit/category_pipe.txt  | 2013-02-08 20:58:27.333472 |      1
 22482 | tickit/date2008_pipe.txt  | 2013-02-08 20:58:28.608305 |      1
 22485 | tickit/allevents_pipe.txt | 2013-02-08 20:58:29.99489  |      1
 22487 | tickit/listings_pipe.txt  | 2013-02-08 20:58:37.632939 |      1
 22489 | tickit/sales_tab.txt      | 2013-02-08 20:58:37.632939 |      1
(6 rows)
```

Validation des données d'entrée

Afin de valider les données des fichiers d'entrée Amazon S3 ou de la table Amazon DynamoDB avant que vous ne chargiez réellement les données, utilisez l'option NOLOAD avec la commande COPY. Utilisez NOLOAD avec les mêmes commandes et options COPY que vous utiliseriez pour charger les données. NOLOAD vérifie l'intégrité de toutes les données sans les charger dans la base de données. L'option NOLOAD affiche les erreurs qui se produisent si vous tentez de charger les données.

Par exemple, si vous avez spécifié un chemin Amazon S3 incorrect pour le fichier d'entrée, Amazon Redshift affiche l'erreur suivante.

```
ERROR:  No such file or directory
DETAIL:
-----------------------------------------------
Amazon Redshift error:  The specified key does not exist
code:      2
context:   S3 key being read :
location:  step_scan.cpp:1883
process:   xenmaster [pid=22199]
-----------------------------------------------
```

Pour résoudre les messages d'erreur, consultez Référence des erreurs de chargement.

Pour obtenir un exemple d'utilisation de l'option NOLOAD, consultez Commande COPY avec l'option NOLOAD.

Chargement des tables avec compression automatique

Vous pouvez appliquer manuellement des codages de compression aux colonnes des tables, en fonction de votre propre évaluation des données. Vous pouvez également utiliser la commande COPY avec COMPUPDATE définie sur ON pour analyser et appliquer automatiquement la compression en fonction des exemples de données.

Vous pouvez utiliser la compression automatique lorsque vous créez et chargez une nouvelle table. La commande COPY effectue une analyse de compression. Vous pouvez également effectuer une analyse de la compression sans charger les données ou en modifiant la compression sur une table en exécutant la commande ANALYZE COMPRESSION sur une table déjà remplie. Par exemple, vous pouvez exécuter ANALYZE COMPRESSION lorsque vous souhaitez analyser la compression

sur une table pour une utilisation ultérieure, tout en préservant les instructions DDL (Data Definition Language) existantes.

La compression automatique équilibre les performances globales lors du choix des encodages de compression. Les analyses à plage restreinte peuvent mal s'exécuter si les colonnes de clé de tri sont beaucoup plus compressées que les autres colonnes de la même requête. Par conséquent, la compression automatique ignore la phase d'analyse des données sur les colonnes de clé de tri et conserve les types d'encodage définis par l'utilisateur.

La compression automatique choisit l'encodage RAW si vous n'avez pas explicitement défini de type d'encodage. ANALYZE COMPRESSION se comporte de la même manière. Pour des performances optimales de requête, envisagez d'utiliser RAW pour les clés de tri.

Fonctionnement de la compression automatique

Lorsque le paramètre COMPUPDATE est défini sur ON, la commande COPY applique la compression automatique chaque fois que vous exécutez la commande COPY avec une table cible vide et que toutes les colonnes de la table ont l'encodage RAW ou aucun codage.

Pour appliquer la compression automatique à une table vide, quel que soit son encodage de compression actuel, exécutez la commande COPY avec l'option COMPUPDATE définie sur ON. Pour désactiver la compression automatique, exécutez la commande COPY avec l'option COMPUPDATE définie sur OFF.

Vous ne pouvez pas appliquer la compression automatique sur une table qui contient déjà des données.

> **Note**
>
> L'analyse de la compression automatique requiert assez de lignes dans les données de chargement (au moins 100 000 lignes par tranche) pour générer un échantillon significatif.

La compression automatique exécute ces opérations en arrière-plan dans le cadre de la transaction de chargement :

1. Un échantillon initial de lignes est chargé depuis le fichier d'entrée. La taille de l'échantillon dépend de la valeur du paramètre COMPROWS. La valeur par défaut est 100 000.
2. Les options de compression sont choisies pour chaque colonne.

3. Les exemples de ligne sont supprimés de la table.
4. La table est recrée avec les encodages de compression choisis.
5. La totalité du fichier en entrée est chargé et compressé à l'aide des nouveaux encodages.

Une fois que vous exécutez la commande COPY, la table est entièrement chargée, compressée et prête à être utilisée. Si vous chargez plus de données ultérieurement, les lignes ajoutées sont compressées selon l'encodage existant.

Si vous souhaitez uniquement effectuer une analyse de la compression, exécutez ANALYZE COMPRESSION, qui est beaucoup plus efficace que l'exécution complète de la commande COPY. Ensuite, vous pouvez évaluer les résultats pour décider d'utiliser la compression automatique ou de recréer manuellement la table.

La compression automatique n'est prise en charge que pour la commande COPY. Vous pouvez aussi appliquer manuellement l'encodage de compression lorsque vous créez la table. Pour plus d'informations sur l'encodage de la compression, consultez Compression de colonnes pour réduire la taille des données stockées.

Exemple de compression automatique

Dans cet exemple, supposons que la base de données TICKIT contienne une copie de la table LISTING appelée BIGLIST et que vous souhaitiez appliquer la compression automatique à cette table lorsqu'elle est chargée avec 3 millions de lignes environ.

Pour charger et compresser automatiquement la table

1. Vérifiez que la table est vide. Vous ne pouvez appliquer la compression automatique qu'à une table vide :

```
TRUNCATE biglist;
```

2. Chargez la table avec une simple commande COPY. Même si la table est vide, un certain encodage antérieur peut avoir été spécifié. Pour permettre à Amazon Redshift d'effectuer une analyse de la compression, définissez le paramètre COMPUPDATE sur ON.

```
COPY biglist FROM 's3://amzn-s3-demo-bucket/biglist.txt'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
DELIMITER '|' COMPUPDATE ON;
```

Comme aucune option COMPROWS n'est spécifiée, la taille de l'échantillon par défaut et recommandée de 100 000 lignes par tranche est utilisée.

3. Regardez le nouveau schéma de la table BIGLIST afin de vérifier les schémas d'encodage choisis automatiquement.

```
SELECT "column", type, encoding
from pg_table_def where tablename = 'biglist';


     Column     |            Type             | Encoding
----------------+-----------------------------+----------
 listid         | integer                     | az64
 sellerid       | integer                     | az64
 eventid        | integer                     | az64
 dateid         | smallint                    | none
 numtickets     | smallint                    | az64
 priceperticket | numeric(8,2)                | az64
 totalprice     | numeric(8,2)                | az64
 listtime       | timestamp without time zone | az64
```

4. Vérifiez que le nombre attendu de lignes a été chargé :

```
select count(*) from biglist;

count
---------
3079952
(1 row)
```

Lorsque les lignes sont ajoutés par la suite à cette table à l'aide des instructions COPY ou INSERT, les mêmes encodages de compression sont appliqués.

Optimisation de stockage pour les tables étroites

Si vous avez une table avec très peu de colonnes, mais un très grand nombre de lignes, les trois colonnes d'identité de métadonnées masquées (INSERT_XID, DELETE_XID, ROW_ID) consomment une quantité disproportionnée de l'espace disque pour la table.

Afin d'optimiser la compression des colonnes masquées, chargez la table en une seule transaction COPY chaque fois que possible. Si vous chargez la table avec plusieurs commandes COPY

distinctes, la colonne INSERT_XID ne compresse pas correctement. Vous devez effectuer une opération VACUUM si vous utilisez plusieurs commandes COPY, mais cela n'améliore pas la compression d'INSERT_XID.

Chargement des valeurs par défaut des colonnes

Vous pouvez le cas échéant définir une liste de colonnes dans votre commande COPY. Si une colonne de la table est omise de la liste des colonnes, COPY charge la colonne avec la valeur fournie par l'option DEFAULT qui a été spécifiée dans la commande CREATE TABLE ou avec NULL si l'option DEFAULT n'a pas été spécifiée.

Si la commande COPY tente d'affecter NULL à une colonne qui est définie comme NOT NULL, elle échoue. Pour plus d'informations sur l'attribution de l'option DEFAULT, consultez CREATE TABLE.

Lors du chargement à partir des fichiers de données sur Amazon S3 les colonnes de la liste des colonnes doivent être dans le même ordre que les champs du fichier de données. Si un champ du fichier de données n'a pas une colonne correspondante dans la liste des colonnes, la commande COPY échoue.

Lors du chargement à partir d'une table Amazon DynamoDB, l'ordre n'importe pas. Tous les champs des attributs Amazon DynamoDB qui ne correspondent pas à une colonne de la table Amazon Redshift sont ignorés.

Les restrictions suivantes s'appliquent lorsque vous utilisez la commande COPY pour charger les valeurs DEFAULT dans une table :

- Si une COLONNE IDENTITY est incluse dans la liste des colonnes, l'option EXPLICIT_IDS doit également être spécifiée dans la commande COPY, sans quoi la commande COPY échoue. De même, si une colonne IDENTITY est absent de la liste des colonnes et que l'option EXPLICIT_IDS est spécifiée, l'opération COPY échoue.
- Comme l'expression DEFAULT évaluée pour une colonne donnée est identique pour toutes les lignes chargées, une expression DEFAULT qui utilise une fonction RANDOM() se verra attribuer la même valeur pour toutes les lignes.
- Les expressions DEFAULT contenant CURRENT_DATE ou SYSDATE sont définies sur l'horodatage de la transaction en cours.

Pour obtenir un exemple, consultez « Charger les données à partir d'un fichier avec les valeurs par défaut » dans Exemples de commandes COPY.

Résolution des problèmes de chargement de données

Lorsque vous chargez des données dans des tables Amazon Redshift, vous pouvez rencontrer des erreurs provenant d'Amazon S3, des données d'entrée non valides et des erreurs de commande COPY. Les sections suivantes fournissent des informations sur l'identification et la résolution des erreurs de chargement de données.

Rubriques

- Résolution des problèmes liés à l'intégration des événements S3 et aux erreurs COPY JOB
- ServiceException Erreurs S3
- Tables système pour la résolution des problèmes de chargement de données
- Erreurs de chargement de caractères multioctets
- Référence des erreurs de chargement

Résolution des problèmes liés à l'intégration des événements S3 et aux erreurs COPY JOB

Utilisez les informations suivantes pour résoudre les problèmes courants liés aux intégrations d'événements Amazon S3 et à COPY JOB avec Amazon Redshift.

Échec de la création de l'intégration de l'événement S3

Si la création de l'intégration des événements S3 a échoué, le statut de l'intégration est`Inactive`. Assurez-vous que les informations suivantes sont correctes pour votre entrepôt de données Amazon Redshift.

- Vous avez ajouté le principal autorisé et la bonne source d'intégration pour votre espace de noms cible dans Amazon Redshift. Consultez Conditions préalables à la création d'une intégration d'événements S3.
- Vous avez ajouté la bonne politique basée sur les ressources au compartiment Amazon S3 source. Consultez Conditions préalables à la création d'une intégration d'événements S3.

Vos données Amazon S3 n'apparaissent pas dans la base de données cible

Si les données d'un COPY JOB n'apparaissent pas, vérifiez les points suivants.

- Interrogez SYS_COPY_JOB_DETAIL pour savoir si le fichier Amazon S3 a été chargé, s'il est en attente d'ingestion ou s'il s'agit d'une erreur. Pour de plus amples informations, veuillez consulter SYS_COPY_JOB_DETAIL.
- Consultez STL_ERROR ou SYS_COPY_JOB_INFO si le fichier Amazon S3 ne s'y trouve pas ou si le temps d'attente est inattendu. Recherchez les erreurs d'identification ou tout élément indiquant que l'intégration est inactive. Pour plus d'informations, consultez STL_ERROR et SYS_COPY_JOB_INFO.

ServiceException Erreurs S3

Les ServiceException erreurs s3 les plus courantes sont causées par une chaîne d'informations d'identification mal formatée ou incorrecte, par le fait que votre cluster et votre compartiment se trouvent dans des AWS régions différentes et par des autorisations Amazon S3 insuffisantes.

La section fournit des informations de dépannage pour chaque type d'erreur.

Chaîne d'informations d'identification non valide

Si votre chaîne d'informations d'identification a été incorrectement formatée, vous recevez le message d'erreur suivant :

```
ERROR: Invalid credentials. Must be of the format: credentials
'aws_access_key_id=<access-key-id>;aws_secret_access_key=<secret-access-key>
[;token=<temporary-session-token>]'
```

Vérifiez que la chaîne d'informations d'identification ne contient pas d'espaces ou de sauts de ligne, et qu'elle est placée entre guillemets simples.

ID de clé d'accès non valide

Si votre ID de clé d'accès n'existe pas, vous recevez le message d'erreur suivant :

```
[Amazon](500310) Invalid operation: S3ServiceException:The AWS Access Key Id you
 provided does not exist in our records.
```

Il s'agit souvent d'une erreur de copier-coller. Vérifiez que l'ID de clé d'accès a été saisi correctement. Si vous utilisez des clés de séance temporaire, vérifiez que la valeur de `token` est définie.

Clé d'accès secrète non valide

Si votre clé d'accès secrète est incorrecte, vous recevrez le message d'erreur suivant :

```
[Amazon](500310) Invalid operation: S3ServiceException:The request signature we
 calculated does not match the signature you provided.
Check your key and signing method.,Status 403,Error SignatureDoesNotMatch
```

Il s'agit souvent d'une erreur de copier-coller. Vérifiez que la clé d'accès secrète a été entrée correctement et qu'il s'agit de la bonne clé pour l'ID de clé d'accès.

Le compartiment est dans une autre région

Le compartiment Amazon S3 spécifié dans la commande COPY doit se trouver dans la même AWS région que le cluster. Si votre compartiment Amazon S3 et votre cluster sont dans des régions différentes, vous recevrez une erreur similaire à celle-ci :

```
ERROR: S3ServiceException:The bucket you are attempting to access must be addressed
 using the specified endpoint.
```

Vous pouvez créer un compartiment Amazon S3 dans une région spécifique en sélectionnant la région lorsque vous créez le compartiment à l'aide de la console de gestion Amazon S3 ou en spécifiant un point de terminaison lorsque vous créez le compartiment à l'aide de l'API ou de la CLI Amazon S3. Pour plus d'informations, consultez Téléchargement de fichiers vers Amazon S3 à utiliser avec COPY.

Pour plus d'informations sur les régions Amazon S3, consultez Accès aux compartiments dans le Guide de l'utilisateur Amazon Simple Storage Service.

Vous pouvez aussi spécifier la région à l'aide de l'option REGION de la commande COPY.

Accès refusé

Si l'utilisateur ne dispose pas d'autorisations suffisantes, vous recevez le message d'erreur suivant :

```
ERROR: S3ServiceException:Access Denied,Status 403,Error AccessDenied
```

L'une des causes possibles est que l'utilisateur identifié par les informations d'identification ne dispose pas d'un accès LIST et GET au compartiment Amazon S3. Pour d'autres causes, consultez Résolution des erreurs d'accès refusé (403 interdit) dans Amazon S3 dans le Guide de l'utilisateur d'Amazon Simple Storage Service.

Pour plus d'informations sur la gestion de l'accès des utilisateurs aux compartiments, consultez Identity and Access Management dans Amazon S3 dans le Guide de l'utilisateur d'Amazon Simple Storage Service.

Tables système pour la résolution des problèmes de chargement de données

Les tables système Amazon Redshift suivantes peuvent être utiles pour le dépannage des problèmes de chargement de données :

- Interrogez STL_LOAD_ERRORS pour découvrir les erreurs qui se sont produites lors de chargements spécifiques.
- Interrogez STL_FILE_SCAN pour afficher les temps de chargement de fichiers spécifiques ou pour voir si un fichier spécifique a été lu.
- Interrogez STL_S3CLIENT_ERROR pour trouver les détails des erreurs rencontrées lors du transfert de données à partir d'Amazon S3.

Pour rechercher et diagnostiquer des erreurs de chargement

1. Créez une vue ou définissez une requête qui renvoie des informations détaillées sur les erreurs de chargement. L'exemple suivant joint la table STL_LOAD_ERRORS à la table STV_TBL_PERM pour associer la table aux noms de table réels. IDs

```
create view loadview as
(select distinct tbl, trim(name) as table_name, query, starttime,
trim(filename) as input, line_number, colname, err_code,
trim(err_reason) as reason
from stl_load_errors sl, stv_tbl_perm sp
where sl.tbl = sp.id);
```

2. Définissez l'option MAXERRORS de votre commande COPY avec une valeur assez grande pour permettre à la commande COPY de renvoyer des informations utiles sur vos données. Si COPY rencontre des erreurs, un message d'erreur vous dirige vers la consultation de la table STL_LOAD_ERRORS pour plus de détails.
3. Interrogez la vue LOADVIEW pour voir les détails de l'erreur. Par exemple :

```
select * from loadview where table_name='venue';
```

```
  tbl   | table_name | query |         starttime
--------+------------+-------+---------------------------
 100551 | venue      | 20974 | 2013-01-29 19:05:58.365391

|     input      | line_number | colname | err_code |       reason
+----------------+-------------+---------+----------+--------------------
| venue_pipe.txt |           1 |       0 |     1214 | Delimiter not found
```

4. Résolvez le problème dans le fichier d'entrée ou le script de chargement, en fonction des informations que la vue renvoie. Certaines erreurs de chargement classique à surveiller particulièrement :

   - Incompatibilité entre les types de données de la table et les valeurs des champs des données d'entrée.
   - Incompatibilité entre le nombre de colonnes de la table et le nombre de champs des données d'entrée.
   - Guillemets non appariés. Amazon Redshift prend en charge les guillemets simples et doubles ; toutefois, ces guillemets doivent être bien équilibrés.
   - Format incorrect pour les date/time données des fichiers d'entrée.
   - Out-of-range valeurs dans les fichiers d'entrée (pour les colonnes numériques).
   - Le nombre de valeurs distinctes pour une colonne dépasse la limite de son encodage de compression.

Erreurs de chargement de caractères multioctets

Les colonnes avec un type de données CHAR acceptent uniquement les caractères UTF-8 codés sur un octet, jusqu'à la valeur d'octet 127, ou la valeur hexadécimale 7F, qui est également le jeu de caractères ASCII. Les colonnes VARCHAR acceptent les caractères UTF-8 multioctets, jusqu'à un maximum de quatre octets. Pour de plus amples informations, veuillez consulter Types caractères.

Si, dans vos données de chargement, une ligne contient un caractère non valide pour le type de données de colonne, COPY renvoie une erreur et enregistre une ligne dans la table du journal système STL_LOAD_ERRORS avec le numéro d'erreur 1220. Le champ ERR_REASON comprend la séquence d'octets, en hexadécimal, pour le caractère non valide.

Au lieu de corriger les caractères non valides dans vos données de chargement, vous pouvez également remplacer les caractères non valides pendant le processus de chargement. Pour remplacer les caractères UTF-8 non valides, spécifiez l'option ACCEPTINVCHARS avec

la commande COPY. Si l'option ACCEPTINVCHARS est définie, le caractère que vous spécifiez remplace le point de code. Si l'option ACCEPTINVCHARS n'est pas définie, Amazon Redshift accepte les caractères comme UTF-8 valides. Pour plus d'informations, consultez ACCEPTINVCHARS.

La liste de points de code suivante est composée d'UTF-8 valides. Les opérations COPY ne renvoient pas d'erreur si l'option ACCEPTINVCHARS n'est pas définie. Toutefois, ces points de code ne sont pas des caractères valides. Vous pouvez utiliser l'option ACCEPTINVCHARS pour remplacer un point de code par un caractère que vous spécifiez. Ces points de code incluent la plage de valeurs de `0xFDD0` à `0xFDEF` et des valeurs allant jusqu'à `0x10FFFF`, se terminant par `FFFE` ou `FFFF` :

- `0xFFFE`, `0x1FFFE`, `0x2FFFE`, ..., `0xFFFFE`, `0x10FFFE`
- `0xFFFF`, `0x1FFFF`, `0x2FFFF`, ..., `0xFFFFF`, `0x10FFFF`

L'exemple suivant affiche le motif de l'erreur lorsque COPY tente de charger le caractère UTF-8 `e0 a1 c7a4` dans une colonne CHAR.

```
Multibyte character not supported for CHAR
(Hint: Try using  VARCHAR). Invalid char: e0 a1 c7a4
```

Si l'erreur est liée à un type de données VARCHAR, le motif de l'erreur inclut un code d'erreur ainsi que la séquence hexadécimale UTF-8 non valide. L'exemple suivant affiche le motif de l'erreur lorsque la commande COPY tente de charger le caractère UTF-8 `a4` dans un champ VARCHAR.

```
String contains invalid or unsupported UTF-8 codepoints.
Bad UTF-8 hex sequence: a4 (error 3)
```

Le tableau suivant répertorie les descriptions et les solutions de contournement suggérées pour les erreurs de chargement VARCHAR. Si l'une de ces erreurs se produit, remplacez le caractère par une séquence de code UTF-8 valide ou supprimez le caractère.

Code d'erreur	Description
1	La séquence d'octets UTF-8 dépasse le maximum de quatre octets pris en charge par VARCHAR.

Code d'erreur	Description
2	La séquence d'octets UTF-8 est incomplète. COPY n'a pas trouvé le nombre d'octets de continuation attendu pour un caractère multioctets avant la fin de la chaîne.
3	Le caractère UTF-8 codé sur un octet est hors de portée. L'octet de départ ne doit pas être 254, 255 ou tout autre caractère entre 128 et 191 (inclus).
4	La valeur de l'octet de fin de la séquence d'octets est hors de portée. L'octet de continuation doit être compris entre 128 et 191 (inclus).
5	Le caractère UTF-8 est réservé comme substitution. Les points de code de substitution (U+D800 à U+DFFF) ne sont pas valides.
8	La séquence d'octets dépasse le point de code UTF-8 maximal.
9	La séquence d'octets UTF-8 n'a pas de point de code correspondant.

Référence des erreurs de chargement

Si des erreurs se produisent lors du chargement des données à partir d'un fichier, interrogez la table STL_LOAD_ERRORS afin d'identifier l'erreur et de déterminer l'explication possible. La table suivante répertorie tous les codes d'erreur qui peuvent se produire au cours des chargements de données :

Codes d'erreur de chargement

Code d'erreur	Description
1200	Erreur d'analyse inconnue. Contactez le support technique.
1201	Délimiteur de champ introuvable dans le fichier d'entrée.
1202	Les données d'entrée ont plus de colonnes qu'il n'était défini dans le DDL.
1203	Les données d'entrée ont moins de colonnes qu'il n'était défini dans le DDL.

Code d'erreur	Description
1204	Les données d'entrée ont dépassé la plage acceptable pour le type de données.
1205	Le format de date n'est pas valide. Consultez Chaînes DATEFORMAT et TIMEFORMAT pour les formats valides.
1206	Le format d'horodatage n'est pas valide. Consultez Chaînes DATEFORMAT et TIMEFORMAT pour les formats valides.
1207	Les données contiennent une valeur en dehors de la plage prévue de 0 à 9.
1208	Erreur de format de type de données FLOAT.
1209	Erreur de format de type de données DECIMAL.
1210	Erreur de format de type de données BOOLEAN.
1211	La ligne en entrée ne contient aucune donnée.
1212	Fichier de chargement introuvable.
1213	Un champ spécifié comme NOT NULL ne contenait aucune donnée.
1214	Délimiteur introuvable.
1215	Erreur de champ CHAR.
1 216	La ligne en entrée n'est pas valide.
1217	La valeur de colonne d'identité n'est pas valide.
1218	Lors de l'utilisation de NULL AS '\0', un champ contenant un indicateur de fin null (NUL ou UTF-8 0000) contenait plus d'un octet.
1219	La valeur hexadécimale UTF-8 contient un chiffre non valide.
1220	La chaîne contient des points de code UTF-8 non valides ou non pris en charge.

Code d'erreur	Description
1221	L'encodage du fichier n'est pas le même que celui spécifié dans la commande COPY.
1222	Erreur de dépassement de valeur entière.
1223	Type de données non valide.
1224	Les données d'entrée ne sont pas bien formées au format JSON ou au type de données SUPER.
8001	COPY avec le paramètre MANIFEST nécessite le chemin d'accès complet d'un objet Amazon S3.
9005	Clé de fin non valide spécifiée.

Créez une intégration d'événements S3 pour copier automatiquement des fichiers depuis des compartiments Amazon S3

> **Note**
>
> La version préliminaire pour la copie automatique est terminée. Par conséquent, les clusters de prévisualisation seront automatiquement supprimés 30 jours après la fin de la période de prévisualisation. Si vous envisagez de continuer à utiliser la copie automatique, nous vous recommandons de recréer vos tâches de copie automatique existantes sur un autre cluster Amazon Redshift. La mise à niveau d'un cluster préliminaire vers la dernière version d'Amazon Redshift n'est pas prise en charge.

Vous pouvez utiliser une tâche de copie automatique pour charger des données dans vos tables Amazon Redshift à partir de fichiers stockés dans Amazon S3. Amazon Redshift détecte lorsque de nouveaux fichiers Amazon S3 sont ajoutés au chemin spécifié dans votre commande COPY. Une commande COPY est ensuite automatiquement exécutée sans que vous ayez à créer un pipeline

d'ingestion de données externe. Amazon Redshift suit les fichiers qui ont été chargés. Amazon Redshift détermine le nombre de fichiers regroupés par commande COPY. Vous pouvez voir les commandes COPY qui en résultent dans les vues système.

La première étape pour créer un COPY JOB automatique consiste à créer une intégration d'événements S3. Lorsqu'un nouveau fichier apparaît dans le compartiment source Amazon S3, Amazon Redshift gère ensuite le chargement des fichiers dans votre base de données à l'aide de la commande COPY.

Conditions préalables à la création d'une intégration d'événements S3

Pour configurer l'intégration de votre événement s3, vérifiez que les conditions préalables suivantes sont remplies.

- Votre compartiment Amazon S3 doit avoir une politique de compartiment qui autorise plusieurs autorisations Amazon S3. Par exemple, l'exemple de politique suivant autorise des autorisations pour le compartiment `amzn-s3-demo-bucket` de ressources hébergé dans*`us-east-1`*. Le compartiment Amazon S3 et l'intégration se trouvent tous deux dans le même boîtier Région AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Auto-Copy-Policy-01",
            "Effect": "Allow",
            "Principal": {
                "Service": "redshift.amazonaws.com"
                },
            "Action": [
                "s3:GetBucketNotification",
                "s3:PutBucketNotification",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringLike": {
                    "aws:SourceArn": "arn:aws:redshift:us-
east-1:123456789012:integration:*",
                    "aws:SourceAccount": "123456789012"
                }
            }
        }
```

```
    ]
}
```

- Votre cluster provisionné Amazon Redshift ou votre espace de noms Redshift Serverless cible doit être autorisé à accéder au compartiment. Vérifiez qu'un rôle IAM associé à votre cluster ou à votre espace de noms sans serveur dispose d'une politique IAM qui autorise les autorisations appropriées. La politique doit autoriser `s3:GetObject` à la fois une ressource de compartiment telle que *amzn-s3-demo-bucket* et `s3:ListBucket` une ressource de compartiment et son contenu tel que*amzn-s3-demo-bucket*/*.

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

Ajoutez votre politique à un rôle IAM doté d'une relation de confiance. Pour ce rôle, procédez comme suit.

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

```
        }
    ]
}
```

Si votre entrepôt de données cible est un cluster provisionné, vous pouvez associer un rôle IAM à votre cluster provisionné à l'aide de la console Amazon Redshift, onglet Autorisations du cluster dans les détails de votre cluster. Pour plus d'informations sur la façon d'associer un rôle à votre cluster provisionné, consultez la section Associer des rôles IAM à des clusters dans le guide de gestion Amazon Redshift.

Si votre entrepôt de données cible est Redshift Serverless, vous pouvez associer un rôle IAM à votre espace de noms sans serveur à l'aide de la console Redshift Serverless, onglet Sécurité et chiffrement dans les détails de votre espace de noms. Pour plus d'informations sur la façon d'associer un rôle à votre espace de noms sans serveur, consultez la section Octroi d'autorisations à Amazon Redshift Serverless dans le guide de gestion Amazon Redshift.

- Votre entrepôt de données Amazon Redshift doit également disposer d'une politique de ressources autorisant le compartiment Amazon S3. Si vous utilisez la console Amazon Redshift, lorsque vous créez l'intégration des événements s3, Amazon Redshift propose l'option Fix it for me permettant d'ajouter cette politique à votre entrepôt de données Amazon Redshift. Pour mettre à jour vous-même une politique de ressources, vous pouvez utiliser la put-resource-policy AWS CLI commande. Par exemple, pour associer une politique de ressources à votre cluster provisionné Amazon Redshift afin d'intégrer un événement S3 à un compartiment Amazon S3, exécutez une AWS CLI commande similaire à la suivante. L'exemple suivant montre une politique pour un espace de noms de cluster provisionné dans le compte *us-east-1* Région AWS for user. *123456789012* Le bucket est nommé*amzn-s3-demo-bucket*.

```
aws redshift put-resource-policy \
--policy file://rs-rp.json \
--resource-arn "arn:aws:redshift: us-east-1:123456789012:namespace/cc4ffe56-
ad2c-4fd1-a5a2-f29124a56433"
```

Où `rs-rp.json` contient :

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

```
   "Principal": {
    "Service": "redshift.amazonaws.com"
   },
   "Action": "redshift:AuthorizeInboundIntegration",
   "Resource": "arn:aws:redshift:us-east-1:123456789012:namespace/cc4ffe56-ad2c-4fd1-
a5a2-f29124a56433",
   "Condition": {
    "StringEquals": {
     "aws:SourceArn": "arn:aws:s3:::amzn-s3-demo-bucket"
    }
   }
  },
  {
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::123456789012:role/myRedshiftRole"
   },
   "Action": "redshift:CreateInboundIntegration",
   "Resource": "arn:aws:redshift:us-east-1:123456789012:namespace/cc4ffe56-ad2c-4fd1-
a5a2-f29124a56433",
   "Condition": {
    "StringEquals": {
     "aws:SourceArn": "arn:aws:s3:::amzn-s3-demo-bucket"
    }
   }
  }
 ]
}
```

Pour associer une politique de ressources à votre espace de noms Redshift Serverless pour l'intégration d'un événement S3 à un compartiment Amazon S3, exécutez une AWS CLI commande similaire à la suivante. L'exemple suivant montre une politique pour un espace de noms sans serveur dans le compte utilisateur *us-east-1* Région AWS for. *123456789012* Le bucket est nommé*amzn-s3-demo-bucket*.

```
aws redshift put-resource-policy \
--policy file://rs-rp.json \
--resource-arn "arn:aws:redshift-serverless:us-east-1:123456789012:namespace/
namespace-1"
```

Où `rs-rp.json` contient :

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Principal": {
    "Service": "redshift.amazonaws.com"
   },
   "Action": "redshift:AuthorizeInboundIntegration",
   "Resource": "arn:aws:redshift-serverless:us-east-1:123456789012:namespace/
namespace-1",
   "Condition": {
    "StringEquals": {
     "aws:SourceArn": "arn:aws:s3:::amzn-s3-demo-bucket"
    }
   }
  },
  {
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::123456789012:user/myUser"
   },
   "Action": "redshift:CreateInboundIntegration",
   "Resource": "arn:aws:redshift-serverless:us-east-1:123456789012:namespace/
namespace-1",
   "Condition": {
    "StringEquals": {
     "aws:SourceArn": "arn:aws:s3:::amzn-s3-demo-bucket"
    }
   }
  }
 ]
}
```

Création d'une intégration d'événements S3

Pour configurer votre tâche de copie, vous devez d'abord définir une intégration d'événements S3.

Amazon Redshift console

Pour créer une intégration d'événements Amazon S3 sur la console Amazon Redshift

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le volet de navigation de gauche, choisissez S3 event integrations.
3. Choisissez Create Amazon S3 event integration pour ouvrir l'assistant de création et choisissez Create Amazon S3 event integration à utiliser avec la copie automatique. Votre compartiment Amazon S3 source et votre entrepôt de données Amazon Redshift cible doivent se trouver dans le même emplacement. Région AWS Spécifiez les informations suivantes lors des étapes de création d'une intégration :

   - Nom de l'intégration — Il s'agit d'un identifiant unique pour toutes les intégrations que vous détenez Compte AWS actuellement Région AWS.
   - Description — Il s'agit d'un texte qui décrit l'intégration des événements Amazon S3 pour référence ultérieure.
   - Compartiment source S3 : le compartiment Amazon S3 se trouve-t-il dans le compartiment actuel Compte AWS et Région AWS est-il à l'origine de l'ingestion de données dans Amazon Redshift ?
   - Entrepôt de données Amazon Redshift : cluster provisionné Amazon Redshift cible ou groupe de travail Redshift Serverless qui reçoit les données issues de l'intégration.

     Si votre Amazon Redshift cible est sur le même compte, vous pouvez sélectionner la cible. Si la cible se trouve dans un autre compte, vous devez spécifier l'ARN de l'entrepôt de données Amazon Redshift. La cible doit disposer d'une politique de ressources avec des principaux autorisés et une source d'intégration. Si vous n'avez pas les bonnes politiques de ressources sur la cible et que celle-ci se trouve dans le même compte, vous pouvez sélectionner l'option Corriger pour moi afin d'appliquer automatiquement les politiques de ressources lors du processus de création d'intégration. Si votre cible se trouve dans une autre région Compte AWS, vous devez appliquer manuellement la politique en matière de ressources à l'entrepôt Amazon Redshift.
4. Entrez jusqu'à 50 clés de balise, avec une valeur facultative : pour fournir des métadonnées supplémentaires sur l'intégration.
5. Une page de révision s'affiche où vous pouvez choisir Créer une intégration d'événements S3.

AWS CLI

Pour créer une intégration d'événements Amazon S3 à l'aide de AWS CLI, utilisez la `create-integration` commande avec les options suivantes :

- `integration-name` : spécifiez le nom de l'intégration.
- `source-arn`— Spécifiez l'ARN du compartiment source Amazon S3.
- `target-arn`— Spécifiez l'ARN de l'espace de noms du cluster provisionné Amazon Redshift ou de la cible du groupe de travail Redshift Serverless.

L'exemple suivant crée une intégration en fournissant le nom de l'intégration, l'ARN source et l'ARN cible. L'intégration n'est pas cryptée.

```
aws redshift create-integration \
--integration-name s3-integration \
--source-arn arn:aws:s3:us-east-1::s3-example-bucket \
--target-arn arn:aws:redshift:us-east-1:123456789012:namespace:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
          {
    "IntegrationArn": "arn:aws:redshift:us-east-1:123456789012:integration:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "IntegrationName": "s3-integration",
    "SourceArn": "arn:aws:s3:::s3-example-bucket",
    "SourceType": "s3-event-notifications",
    "TargetArn": "arn:aws:redshift:us-east-1:123456789012:namespace:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "Status": "creating",
    "Errors": [],
    "CreateTime": "2024-10-09T19:08:52.758000+00:00",
    "Tags": []
}
```

Vous pouvez également utiliser les AWS CLI commandes suivantes pour gérer l'intégration de vos événements S3.

- `delete-integration`— Spécifiez un ARN d'intégration pour supprimer une intégration d'événements S3.
- `modify-integration`— Spécifiez un ARN d'intégration pour modifier le nom ou la description (ou les deux) d'une intégration d'événements S3.

- `describe-integrations`— Spécifiez un ARN d'intégration pour afficher les propriétés d'une intégration d'événements S3.

Consultez le guide de la CLI Amazon Redshift pour plus d'informations sur ces commandes.

Amazon Redshift crée ensuite une intégration d'événements S3 avec la source et la cible associées, le statut et les informations relatives à l'état d'une tâche de copie automatique associée. Vous pouvez consulter les informations relatives à l'intégration d'un événement S3 sur la console Amazon Redshift en choisissant les intégrations d'événements S3 et en choisissant l'intégration pour en afficher les détails. Les intégrations sont séparées par celles créées dans mon compte et celles créées dans d'autres comptes. La liste Dans mon compte indique les intégrations où la source et la cible se trouvent dans le même compte. La liste Depuis d'autres comptes indique les intégrations dans lesquelles la source appartient à un autre compte.

Si vous supprimez une intégration d'événements S3, le statut COPY JOB correspondant passe de 1 (actif) à `0` (inactif/en attente). Cependant, le COPY JOB correspondant n'est pas automatiquement supprimé. Si vous essayez ultérieurement de créer un COPY JOB portant le même nom, il se peut qu'il y ait un conflit.

Création et suivi d'un COPY JOB

Une fois l'intégration créée, sur la page des détails de l'intégration des événements S3 correspondant à l'intégration que vous avez créée, choisissez Create autocopy job pour accéder à l'éditeur de requêtes Amazon Redshift v2 où vous pouvez créer la tâche de copie automatique pour l'intégration. Amazon Redshift fait correspondre le compartiment de la clause FROM de l'instruction COPY JOB CREATE au compartiment utilisé pour l'intégration des événements S3. Pour plus d'informations sur l'utilisation de l'éditeur de requêtes Amazon Redshift v2, consultez la section Interrogation d'une base de données à l'aide de l'éditeur de requêtes Amazon Redshift v2 dans le guide de gestion Amazon Redshift. Par exemple, exécutez la commande COPY suivante dans l'éditeur de requêtes v2 pour créer un COPY JOB automatique qui associe le compartiment Amazon S3 `s3://amzn-s3-demo-bucket/staging-folder` à une intégration d'événements Amazon S3.

```
COPY public.target_table
FROM 's3://amzn-s3-demo-bucket/staging-folder'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyLoadRoleName'
JOB CREATE my_copy_job_name
AUTO ON;
```

Vous définissez une COPY JOB une seule fois. Les mêmes paramètres sont utilisés pour les future exécutions.

Pour définir et gérer un COPY JOB, vous devez disposer d'une autorisation. Pour plus d'informations sur l'octroi et la révocation d'une autorisation sur un COPY JOB, consultez GRANT etREVOKE. Pour plus d'informations sur l'octroi et la révocation d'autorisations étendues pour un COPY JOB, consultez Octroi d’autorisations étendues et. Révocation d’autorisations étendues

Vous gérez les opérations de chargement à l’aide des options CREATE, LIST, SHOW, DROP, ALTER et RUN des tâches. Pour de plus amples informations, veuillez consulter TÂCHE DE COPIE.

Vous pouvez interroger les vues du système pour voir le statut et la progression de COPY JOB. Les vues sont fournies comme suit :

- SYS_COPY_JOB – contient une ligne pour chaque COPY JOB actuellement définie.
- SYS_COPY_JOB_DETAIL— contient des informations sur les fichiers en attente, en erreur et ingérés pour chaque COPY JOB.
- SYS_COPY_JOB_INFO— contient les messages enregistrés à propos d'un COPY JOB.
- SYS_LOAD_HISTORY – contient des informations détaillées sur les commandes COPY.
- SYS_LOAD_ERROR_DETAIL – contient des informations détaillées sur les erreurs de commande COPY.
- SVV_COPY_JOB_INTEGRATIONS— contient des informations détaillées sur les intégrations d'événements S3.
- STL_LOAD_ERRORS – contient des erreurs provenant des commandes COPY.
- STL_LOAD_COMMITS – contient des informations permettant de résoudre les problèmes liés au chargement des données d’une commande COPY.

Pour plus d'informations sur la résolution des erreurs d'intégration des événements S3, consultezRésolution des problèmes liés à l'intégration des événements S3 et aux erreurs COPY JOB.

Pour obtenir la liste des fichiers chargés par un COPY JOB, exécutez le code SQL suivant, mais remplacez d'abord *<job_id>* :

```
SELECT job_id, job_name, data_source, copy_query, filename, status, curtime
FROM sys_copy_job copyjob
JOIN stl_load_commits loadcommit
ON copyjob.job_id = loadcommit.copy_job_id
```

```
WHERE job_id = <job_id>;
```

Considérations relatives à la création d'une intégration d'événements S3 pour la copie automatique

Tenez compte des points suivants lorsque vous utilisez la copie automatique.

- Vous pouvez créer un maximum de 200 TÂCHES DE COPIE pour chaque cluster ou groupe de travail d'un Compte AWS.
- Vous pouvez créer un maximum de 50 intégrations d'événements S3 pour chaque cible Amazon Redshift.
- Vous ne pouvez pas créer d'intégration d'événements S3 avec un compartiment Amazon S3 source dont le nom du compartiment comporte un point (.).
- Vous ne pouvez créer qu'une seule intégration d'événements S3 entre la même source et la même cible. En d'autres termes, il ne peut y avoir qu'une seule intégration d'événements S3 entre un compartiment Amazon S3 et un entrepôt de données Amazon Redshift à la fois.
- Vous ne pouvez pas avoir de notifications d'événement existantes pour le type `S3_OBJECT_CREATED` d'événement défini dans le compartiment Amazon S3 source. Cependant, une fois l'intégration d'un événement S3 créée, vous pouvez mettre à jour la notification d'événement du compartiment Amazon S3 prefix/suffix avec une portée plus étroite. De cette façon, vous pouvez également effectuer une configuration `S3_OBJECT_CREATED` prefix/suffix pour une autre cible et éviter tout conflit avec l'intégration des événements S3. Si vous rencontrez des problèmes liés au fait que la copie automatique ne s'exécute pas comme prévu, préparez le AWS CloudTrail journal de l'`s3:PutBucketNotificationConfiguration`action sur votre compartiment S3 pour la période en question lorsque vous contactez AWS Support.

Chargement de tables à l'aide de commandes DML

Amazon Redshift prend en charge les commandes standard du langage de manipulation de données (INSERT, UPDATE et DELETE), que vous pouvez utiliser pour modifier les lignes des tables. Vous pouvez aussi utiliser la commande TRUNCATE pour des suppressions en bloc rapides.

> **Note**
>
> Nous vous encourageons vivement à utiliser la commande COPY pour charger de grandes quantités de données. La lenteur liée à l'utilisation d'instructions INSERT pour remplir une

table peut être prohibitive. Sinon, si vos données existent déjà dans d'autres tables de bases de données Amazon Redshift, utilisez INSERT INTO ... SELECT FROM ou CREATE TABLE AS pour améliorer les performances. Pour plus d'informations, consultez INSERT ou CREATE TABLE AS.

Si vous insérez, mettez à jour ou supprimez un nombre important de lignes dans une table, par rapport au nombre de lignes avant les modifications, exécutez les commandes ANALYZE et VACUUM sur la table lorsque vous avez terminé. Si un certain nombre de petites modifications s'accumulent au fil du temps dans votre application, vous souhaiterez peut-être planifier les commandes ANALYZE et VACUUM pour qu'elles s'exécutent à intervalles réguliers. Pour plus d'informations, consultez Analyse des tables et Exécution de l'opération VACUUM sur les tables.

Rubriques

- Mise à jour et insertion de nouvelles données

Mise à jour et insertion de nouvelles données

Vous pouvez ajouter de manière efficace de nouvelles données à une table existante à l'aide de la commande MERGE. Effectuez une opération de fusion en créant une table intermédiaire, puis utilisez l'une des méthodes décrites dans cette section pour mettre à jour la table cible à partir de la table intermédiaire. Pour en savoir plus sur la commande MERGE, consultez MERGE.

Exemples de fusion utilise un exemple de jeu de données pour Amazon Redshift, appelé jeu de données TICKIT. Comme condition préalable, vous pouvez configurer les tables et les données TICKIT en suivant les instructions disponibles dans Démarrage avec les tâches courantes liées aux bases de données. Des informations plus détaillées sur le jeu de données en exemple sont disponibles dans la base de données en exemple.

Méthode de fusion 1 : remplacement des lignes existantes

Si vous remplacez toutes les colonnes de la table cible, le moyen le plus rapide d'effectuer une fusion est de remplacer les lignes existantes. La table cible est analysée une seule fois, et une jointure interne est utilisée pour supprimer les lignes qui sont appelées à être mises à jour. Une fois que les lignes ont été supprimées, elles sont remplacées par de nouvelles lignes via une opération d'insertion unique à partir de la table intermédiaire.

Utilisez cette méthode si toutes les conditions suivantes sont vraies :

- Votre table cible votre table intermédiaire contiennent les mêmes colonnes.
- Vous avez l'intention de remplacer toutes les données des colonnes de la table cible par toutes les colonnes de la table intermédiaire.
- Vous allez utiliser toutes les lignes de la table intermédiaire dans la fusion.

Si l'un de ces critères ne s'applique pas, utilisez la méthode de fusion 2 : spécification d'une liste de colonnes sans utiliser MERGE, décrite dans la section suivante.

Si vous n'utilisez pas toutes les lignes de la table intermédiaire, filtrez les instructions DELETE et INSERT en utilisant une clause WHERE pour ignorer les lignes qui ne sont pas modifiées. Cependant, si la plupart des lignes de la table intermédiaire ne prennent pas part à la fusion, nous recommandons d'exécuter les commandes UPDATE et INSERT en étapes distinctes, comme décrit ultérieurement dans cette section.

Méthode de fusion 2 : spécification d'une liste de colonnes sans utiliser MERGE

Utilisez cette méthode pour mettre à jour des colonnes spécifiques de la table cible au lieu de remplacer des lignes entières. Cette méthode prend plus de temps que la méthode précédente, car elle nécessite une étape de mise à jour supplémentaire et elle n'utilise pas la commande MERGE. Utilisez cette méthode si l'une des conditions suivantes est vraie :

- Certaines colonnes de la table cible ne doivent pas être mises à jour.
- La plupart des lignes de la table intermédiaire ne sont pas utilisées dans les mises à jour.

Rubriques

- Création d'une table temporaire intermédiaire
- Exécution d'une opération de fusion par remplacement des lignes existantes
- Exécution d'une opération de fusion en spécifiant une liste de colonnes sans utiliser la commande MERGE
- Exemples de fusion

Création d'une table temporaire intermédiaire

La table intermédiaire est une table temporaire qui contient toutes les données qui seront utilisées pour apporter des modifications à la table cible, mises à jour et insertions incluses.

Une opération de fusion nécessite une jointure entre la table intermédiaire et la table cible. Pour rassembler les lignes de la jointure, définissez la clé de distribution de la table intermédiaire avec la même colonne que la clé de distribution de la table cible. Par exemple, si la table cible utilise une colonne de clé étrangère comme clé de distribution, utilisez la même colonne pour la clé de distribution de la table intermédiaire. Si vous créez la table intermédiaire à l'aide d'une instruction CREATE TABLE LIKE, la table intermédiaire hérite de la clé de distribution de la table parent. Si vous utilisez une instruction CREATE TABLE AS, la nouvelle table n'hérite pas de la clé de distribution. Pour plus d'informations, consultez Distribution des données pour l'optimisation des requêtes

Si la clé de distribution n'est pas la même que la clé primaire et que la clé de distribution n'est pas mise à jour dans le cadre de l'opération de fusion, ajoutez un prédicat de jointure redondant sur les colonnes de clé de distribution pour permettre une jointure colocalisée. Par exemple :

```
where target.primarykey = stage.primarykey
and target.distkey = stage.distkey
```

Pour vérifier que la requête utilise une jointure colocalisée, exécutez la requête avec EXPLAIN et recherchez DS_DIST_NONE sur l'ensemble des jointures. Pour plus d'informations, consultez Évaluation du plan de requête

Exécution d'une opération de fusion par remplacement des lignes existantes

Exécutez la totalité de l'opération de fusion détaillée dans la procédure, sauf la création et la suppression de la table intermédiaire temporaire, dans une même transaction. La transaction est annulée en cas d'échec d'une étape. L'utilisation d'une seule transaction réduit aussi le nombre de validations, ce qui permet de gagner du temps et d'économiser des ressources.

Pour exécuter une opération de fusion par remplacement des lignes existantes

1. Créez une table intermédiaire, puis complétez-la avec les données à fusionner, comme illustré dans le pseudocode suivant.

   ```
   CREATE temp table stage (like target);

   INSERT INTO stage
   SELECT * FROM source
   WHERE source.filter = 'filter_expression';
   ```

2. Utilisez MERGE pour effectuer une jointure interne avec la table intermédiaire afin de mettre à jour les lignes de la table cible qui ont une correspondance dans la table intermédiaire, puis

insérez toutes les lignes restantes dans la table cible qui n'ont pas de correspondance dans la table intermédiaire.

Nous vous recommandons d'exécuter les opérations de mise à jour et d'insertion dans une même commande MERGE.

```
MERGE INTO target
USING stage [optional alias] on (target.primary_key = stage.primary_key)
WHEN MATCHED THEN
UPDATE SET col_name1 = stage.col_name1 , col_name2= stage.col_name2, col_name3 =
 {expr}
WHEN NOT MATCHED THEN
INSERT (col_name1 , col_name2, col_name3) VALUES (stage.col_name1, stage.col_name2,
 {expr});
```

3. Supprimez la table intermédiaire.

```
DROP TABLE stage;
```

Exécution d'une opération de fusion en spécifiant une liste de colonnes sans utiliser la commande MERGE

Exécutez la totalité de l'opération de fusion détaillée dans la procédure dans une même transaction. La transaction est annulée en cas d'échec d'une étape. L'utilisation d'une seule transaction réduit aussi le nombre de validations, ce qui permet de gagner du temps et d'économiser des ressources.

Pour exécuter une opération de fusion par spécification d'une liste de colonnes

1. Placez l'ensemble de l'opération dans un seul bloc de transactions.

```
BEGIN transaction;
…
END transaction;
```

2. Créez une table intermédiaire, puis complétez-la avec les données à fusionner, comme illustré dans le pseudocode suivant.

```
create temp table stage (like target);
insert into stage
select * from source
```

```
where source.filter = 'filter_expression';
```

3. Mettez à jour la table cible à l'aide d'une jointure interne avec la table intermédiaire.

   - Dans la clause UPDATE, listez explicitement les colonnes à mettre à jour.
   - Effectuez une jointure interne avec la table intermédiaire.
   - Si la clé de distribution est différente de la clé primaire et que la clé de distribution n'est pas mise à jour, ajoutez une jointure redondante sur la clé de distribution. Pour vérifier que la requête utilise une jointure colocalisée, exécutez la requête avec EXPLAIN et recherchez DS_DIST_NONE sur l'ensemble des jointures. Pour plus d'informations, consultez Évaluation du plan de requête
   - Si votre table cible est triée par horodatage, ajoutez un prédicat pour tirer parti des analyses à plage restreinte sur la table cible. Pour de plus amples informations, veuillez consulter Bonnes pratiques Amazon Redshift pour la conception de requêtes.
   - Si vous n'utilisez pas toutes les lignes de la fusion, ajoutez une clause pour filtrer les lignes que vous ne voulez pas modifier. Par exemple, ajoutez un filtre d'inégalité sur une ou plusieurs colonnes afin d'exclure les lignes qui n'ont pas changé.
   - Placez les opérations de mise à jour, de suppression et d'insertion dans un même bloc de transaction de telle sorte qu'en cas de problème, tout puisse être annulé.

   Par exemple :

```
begin transaction;

update target
set col1 = stage.col1,
col2 = stage.col2,
col3 = 'expression'
from stage
where target.primarykey = stage.primarykey
and target.distkey = stage.distkey
and target.col3 > 'last_update_time'
and (target.col1 != stage.col1
or target.col2 != stage.col2
or target.col3 = 'filter_expression');
```

4. Supprimez les lignes superflues de la table intermédiaire à l'aide d'une jointure interne avec la table cible. Certaines lignes de la table cible correspondent déjà aux lignes correspondantes de

la table intermédiaire, tandis que d'autres ont été mises à jour lors de l'étape précédente. Dans les deux cas, elles ne sont pas nécessaires pour l'insertion.

```
delete from stage
using target
where stage.primarykey = target.primarykey;
```

5. Insérez les lignes restantes de la table intermédiaire. Utilisez la même liste de la colonne dans la clause VALUES que celle que vous avez utilisée dans l'instruction UPDATE à l'étape 2.

```
insert into target
(select col1, col2, 'expression'
from stage);

end transaction;
```

6. Supprimez la table intermédiaire.

```
drop table stage;
```

Exemples de fusion

Les exemples suivants effectuent une fusion pour mettre à jour la table SALES. Le premier exemple utilise la méthode la plus simple pour supprimer les lignes de la table cible, puis insérer toutes les lignes de la table intermédiaire. Comme le deuxième exemple nécessite une mise à jour des colonnes de la sélection, elle inclut une étape de mise à jour supplémentaire.

Exemples de fusion utilise un exemple de jeu de données pour Amazon Redshift, appelé jeu de données TICKIT. Comme condition préalable, vous pouvez configurer les tables et les données TICKIT en suivant les instructions disponibles dans le guide Démarrage avec les tâches courantes liées aux bases de données. Des informations plus détaillées sur le jeu de données en exemple sont disponibles dans la base de données en exemple.

Exemple de source de données de fusion

Les exemples de cette section nécessitent un exemple de source de données qui inclut les mises à jour et les insertions. Pour les exemples, nous allons créer un exemple de table nommée SALES_UPDATE qui utilise les données de la table SALES. Nous allons remplir la nouvelle table avec des données aléatoires qui représentant les nouvelles activités de vente pour décembre.

Nous allons utiliser l'exemple de table SALES_UPDATE pour créer la table intermédiaire dans les exemples suivants.

```
-- Create a sample table as a copy of the SALES table.

create table tickit.sales_update as
select * from tickit.sales;

-- Change every fifth row to have updates.

update tickit.sales_update
set qtysold = qtysold*2,
pricepaid = pricepaid*0.8,
commission = commission*1.1
where saletime > '2008-11-30'
and mod(sellerid, 5) = 0;

-- Add some new rows to have inserts.
-- This example creates a duplicate of every fourth row.

insert into tickit.sales_update
select (salesid + 172456) as salesid, listid, sellerid, buyerid, eventid, dateid,
 qtysold, pricepaid, commission, getdate() as saletime
from tickit.sales_update
where saletime > '2008-11-30'
and mod(sellerid, 4) = 0;
```

Exemple de fusion qui remplace les lignes existantes sur la base de clés correspondantes

Le script suivant utilise la table SALES_UPDATE pour effectuer une opération de fusion sur la table SALES avec de nouvelles données pour l'activité de vente de décembre. Cet exemple remplace les lignes de la table SALES qui ont été mises à jour. Pour cet exemple, nous allons mettre à jour les colonnes qtysold et pricepaid, mais laisser commission et saletime inchangées.

```
MERGE into tickit.sales
USING tickit.sales_update sales_update
on ( sales.salesid = sales_update.salesid
and sales.listid = sales_update.listid
and sales_update.saletime > '2008-11-30'
and (sales.qtysold != sales_update.qtysold
or sales.pricepaid != sales_update.pricepaid))
WHEN MATCHED THEN
```

```
update SET qtysold = sales_update.qtysold,
pricepaid = sales_update.pricepaid
WHEN NOT MATCHED THEN
INSERT (salesid, listid, sellerid, buyerid, eventid, dateid, qtysold , pricepaid,
 commission, saletime)
values (sales_update.salesid, sales_update.listid, sales_update.sellerid,
 sales_update.buyerid, sales_update.eventid,
sales_update.dateid, sales_update.qtysold , sales_update.pricepaid,
 sales_update.commission, sales_update.saletime);

-- Drop the staging table.
drop table tickit.sales_update;

-- Test to see that commission and salestime were not impacted.
SELECT sales.salesid, sales.commission, sales.salestime, sales_update.commission,
 sales_update.salestime
FROM tickit.sales
INNER JOIN tickit.sales_update sales_update
ON
sales.salesid = sales_update.salesid
AND sales.listid = sales_update.listid
AND sales_update.saletime > '2008-11-30'
AND (sales.commission != sales_update.commission
OR sales.salestime != sales_update.salestime);
```

Exemple de fusion qui spécifie une liste de colonnes sans utiliser MERGE

L'exemple suivant effectue une opération de fusion pour mettre à jour SALES avec de nouvelles données pour l'activité de vente de décembre. Nous avons besoin d'exemples de données qui incluent les opérations de mise à jour et d'insertion, ainsi que les lignes qui n'ont pas changé. Pour cet exemple, nous voulons mettre à jour les colonnes QTYSOLD et PRICEPAID, mais laisser COMMISSION et SALETIME telles quelles. Le script suivant utilise la table SALES_UPDATE pour effectuer une opération de fusion sur la table SALES.

```
-- Create a staging table and populate it with rows from SALES_UPDATE for Dec
create temp table stagesales as select * from sales_update
where saletime > '2008-11-30';

-- Start a new transaction
begin transaction;

-- Update the target table using an inner join with the staging table
```

```
-- The join includes a redundant predicate to collocate on the distribution key -- A
 filter on saletime enables a range-restricted scan on SALES

update sales
set qtysold = stagesales.qtysold,
pricepaid = stagesales.pricepaid
from stagesales
where sales.salesid = stagesales.salesid
and sales.listid = stagesales.listid
and stagesales.saletime > '2008-11-30'
and (sales.qtysold != stagesales.qtysold
or sales.pricepaid != stagesales.pricepaid);

-- Delete matching rows from the staging table
-- using an inner join with the target table

delete from stagesales
using sales
where sales.salesid = stagesales.salesid
and sales.listid = stagesales.listid;

-- Insert the remaining rows from the staging table into the target table
insert into sales
select * from stagesales;

-- End transaction and commit
end transaction;

-- Drop the staging table
drop table stagesales;
```

Exécution d’une copie complète

Une copie recrée et remplit une table à l’aide d’une insertion en bloc, qui trie automatiquement la table. Si une table possède une grande région non triée, une copie complète est beaucoup plus rapide qu’une opération VACUUM. Nous vous recommandons d’effectuer des mises à jour simultanées dans le cadre d’une opération de copie complète seulement si vous pouvez en assurer le suivi. Une fois le processus terminé, déplacez les mises à jour delta dans la nouvelle table. Une opération VACUUM prend automatiquement en charge les mises à jour simultanées.

Vous pouvez choisir l’une des quatre méthodes pour créer une copie de la table d’origine :

- Utilisez la table DDL d'origine.

  Si la commande CREATE TABLE DDL est disponible, il s'agit de la méthode préférée et la plus rapide. Si vous créez une nouvelle table, vous pouvez spécifier tous les attributs de table et de colonne, y compris la clé primaire et les clés étrangères. Vous pouvez trouver le DDL d'origine à l'aide de la fonction SHOW TABLE.
- Utilisez CREATE TABLE LIKE.

  Si le DDL d'origine n'est pas disponible, vous pouvez utiliser CREATE TABLE LIKE pour recréer la table d'origine. La nouvelle table hérite des attributs encoding, distribution key, sort key et not-null de la table parent. La nouvelle table n'hérite pas de la clé primaire et des attributs de clés étrangères de la table parent, mais vous pouvez les ajouter avec ALTER TABLE.
- Créez une table temporaire et tronquez la table d'origine.

  Si vous devez conserver les attributs de clé primaire et de clé étrangère de la table parent. Si la table parent présente des dépendances, vous pouvez utiliser CREATE TABLE ... AS (CTAS) pour créer une table temporaire. Tronquez ensuite la table d'origine et remplissez-la à partir de la table temporaire.

  L'utilisation d'une table temporaire améliore les performances de façon significative par rapport à l'utilisation d'une table permanente, mais il existe un risque de perte des données. Une table temporaire est automatiquement supprimée à la fin de la séance dans laquelle elle a été créée. TRUNCATE valide immédiatement, même à l'intérieur d'un bloc de transaction. Si l'opération TRUNCATE réussit, mais que la session s'arrête avant la fin de l'opération INSERT suivante, les données sont perdues. Si la perte de données n'est pas acceptable, utilisez une table permanente.

Après avoir créé une copie d'une table, vous pouvez être amené à accorder l'accès à la nouvelle table. Vous pouvez utiliser GRANT pour définir les privilèges d'accès. Pour consulter et octroyer tous les privilèges d'accès d'une table, vous devez être l'une des personnes suivantes :

- Un super-utilisateur.
- Le propriétaire de la table que vous souhaitez copier.
- Un utilisateur disposant du privilège ACCESS SYSTEM TABLE pour consulter les privilèges de la table et du privilège d'octroi pour toutes les autorisations pertinentes.

En outre, vous pouvez être amené à octroyer une autorisation d'utilisation pour le schéma dans lequel se trouve votre copie complète. L'octroi d'une autorisation d'utilisation est nécessaire si le

schéma de votre copie complète est différent du schéma de la table d'origine, et s'il ne s'agit pas non plus du schéma `public`. Pour consulter et octroyer des privilèges d'utilisation, vous devez être l'une des personnes suivantes :

- Un super-utilisateur.
- Un utilisateur pouvant octroyer l'autorisation USAGE pour le schéma de la copie complète.

Pour exécuter une copie complète à l'aide de la table DDL d'origine

1. (Facultatif) Recréez la table DDL en exécutant un script appelé `v_generate_tbl_ddl`.
2. Créer une copie de la table à l'aide du CREATE TABLE DDL d'origine.
3. Utilisez une instruction INSERT INTO … SELECT pour remplir la copie avec les données de la table d'origine.
4. Vérifiez les autorisations octroyées pour l'ancienne table. Vous pouvez trouver ces autorisations dans la vue système SVV_RELATION_PRIVILEGES.
5. Si nécessaire, accordez les autorisations de l'ancienne table à la nouvelle.
6. Accordez l'autorisation d'utilisation à tous les groupes et utilisateurs qui disposent de privilèges dans la table d'origine. Cette étape n'est pas nécessaire si la table de la copie complète se trouve dans le schéma `public` ou dans le même schéma que la table d'origine.
7. Supprimez la table d'origine.
8. Utilisez une instruction ALTER TABLE pour renommer la copie avec le nom d'origine de la table.

L'exemple suivant effectue une copie complète de la table SAMPLE en utilisant un double de SAMPLE nommé sample_copy.

```
--Create a copy of the original table in the sample_namespace namespace using the
 original CREATE TABLE DDL.
create table sample_namespace.sample_copy ( … );

--Populate the copy with data from the original table in the public namespace.
insert into sample_namespace.sample_copy (select * from public.sample);

--Check SVV_RELATION_PRIVILEGES for the original table's privileges.
select * from svv_relation_privileges where namespace_name = 'public' and relation_name
 = 'sample' order by identity_type, identity_id, privilege_type;

--Grant the original table's privileges to the copy table.
```

```
grant DELETE on table sample_namespace.sample_copy to group group1;
grant INSERT, UPDATE on table sample_namespace.sample_copy to group group2;
grant SELECT on table sample_namespace.sample_copy to user1;
grant INSERT, SELECT, UPDATE on table sample_namespace.sample_copy to user2;

--Grant usage permission to every group and user that has privileges in the original
 table.
grant USAGE on schema sample_namespace to group group1, group group2, user1, user2;

--Drop the original table.
drop table public.sample;

--Rename the copy table to match the original table's name.
alter table sample_namespace.sample_copy rename to sample;
```

Pour effectuer une copie complète à l'aide de CREATE TABLE LIKE

1. Créez une table à l'aide de CREATE TABLE LIKE.
2. Utilisez une instruction INSERT INTO … SELECT pour copier les lignes de la table en cours vers la nouvelle table.
3. Vérifiez les autorisations octroyées pour l'ancienne table. Vous pouvez trouver ces autorisations dans la vue système SVV_RELATION_PRIVILEGES.
4. Si nécessaire, accordez les autorisations de l'ancienne table à la nouvelle.
5. Accordez l'autorisation d'utilisation à tous les groupes et utilisateurs qui disposent de privilèges dans la table d'origine. Cette étape n'est pas nécessaire si la table de la copie complète se trouve dans le schéma `public` ou dans le même schéma que la table d'origine.
6. Supprimez la table actuelle.
7. Utilisez une instruction ALTER TABLE pour renommer la nouvelle table avec le nom d'origine de la table.

L'exemple suivant effectue une copie complète de la table SAMPLE à l'aide de CREATE TABLE LIKE.

```
--Create a copy of the original table in the sample_namespace namespace using CREATE
 TABLE LIKE.
create table sameple_namespace.sample_copy (like public.sample);

--Populate the copy with data from the original table.
insert into sample_namespace.sample_copy (select * from public.sample);
```

```
--Check SVV_RELATION_PRIVILEGES for the original table's privileges.
select * from svv_relation_privileges where namespace_name = 'public' and relation_name
 = 'sample' order by identity_type, identity_id, privilege_type;

--Grant the original table's privileges to the copy table.
grant DELETE on table sample_namespace.sample_copy to group group1;
grant INSERT, UPDATE on table sample_namespace.sample_copy to group group2;
grant SELECT on table sample_namespace.sample_copy to user1;
grant INSERT, SELECT, UPDATE on table sample_namespace.sample_copy to user2;

--Grant usage permission to every group and user that has privileges in the original
 table.
grant USAGE on schema sample_namespace to group group1, group group2, user1, user2;

--Drop the original table.
drop table public.sample;

--Rename the copy table to match the original table's name.
alter table sample_namespace.sample_copy rename to sample;
```

Pour effectuer une copie en créant une table temporaire et en tronquant la table d'origine

1. Utilisez CREATE TABLE AS pour créer une table temporaire avec les lignes de la table d'origine.
2. Tronquez la table en cours.
3. Utilisez une instruction INSERT INTO … SELECT pour copier les lignes de la table temporaire vers la table d'origine.
4. Suppression de la table temporaire.

L'exemple suivant effectue une copie complète de la table SALES en créant une table temporaire et en tronquant la table d'origine. Comme la table d'origine est conservée, vous n'avez pas besoin d'accorder d'autorisations à la table de copie.

```
--Create a temp table copy using CREATE TABLE AS.
create temp table salestemp as select * from sales;

--Truncate the original table.
truncate sales;

--Copy the rows from the temporary table to the original table.
insert into sales (select * from salestemp);
```

```
--Drop the temporary table.
drop table salestemp;
```

Analyse des tables

L'opération ANALYZE met à jour les métadonnées statistiques que le planificateur de requête utilise pour choisir les plans optimaux.

Dans la plupart des cas, vous n'avez pas besoin d'exécuter explicitement la commande ANALYZE. Amazon Redshift surveille les modifications apportées à votre application et met à jour automatiquement les statistiques en arrière-plan. De plus, la commande COPY effectue automatiquement une analyse lorsqu'elle charge des données dans une table vide.

Pour analyser explicitement une table ou la base de données complète, exécutez la commande ANALYSE.

Analyse automatique

Amazon Redshift surveille en permanence votre base de données et effectue automatiquement des opérations d'analyse en arrière-plan. Pour réduire au maximum l'impact sur vos performances système, l'analyse automatique s'exécute pendant les périodes où les charges de travail sont légères.

L'analyse automatique est activée par défaut. Pour désactiver l'analyse automatique, attribuez au paramètre `auto_analyze` la valeur **`false`** en modifiant le groupe de paramètres de votre cluster.

Pour réduire le temps de traitement et améliorer les performances globales du système, Amazon Redshift ignore l'analyse automatique pour toute table dont l'ampleur des modifications est petite.

Une opération d'analyse ignore les tables contenant des up-to-date statistiques. Si vous exécutez ANALYZE dans le cadre de votre flux de travail ETL (extraction, transformation et chargement), l'analyse automatique ignore les tables dont les statistiques sont à jour. De façon similaire, une opération ANALYZE explicite ignore les tables pour lesquelles l'analyse automatique a mis à jour les statistiques.

Analyse des données des nouvelles tables

Par défaut, la commande COPY effectue une opération ANALYZE après avoir chargé des données dans une table vide. Vous pouvez forcer une opération ANALYZE, indépendamment du fait

qu'une table soit vide ou non, en définissant STATUPDATE avec la valeur ON. Si vous spécifiez STATUPDATE OFF, l'opération ANALYZE n'est pas effectuée. Seul le propriétaire de la table ou un super-utilisateur peut exécuter la commande ANALYZE ou la commande COPY avec STATUPDATE ayant ON comme valeur.

Amazon Redshift analyse également les nouvelles tables que vous créez avec les commandes suivantes :

- CREATE TABLE AS (CTAS)
- CREATE TEMP TABLE AS
- SELECT INTO

Amazon Redshift renvoie un message d'avertissement lorsque vous exécutez une requête sur une nouvelle table qui n'a pas été analysée après le chargement initial de ses données. Aucun avertissement n'est fourni lorsque vous interrogez une table après un chargement ou une mise à jour ultérieur(e). Le même message d'avertissement est renvoyé lorsque vous exécutez la commande EXPLAIN sur une requête qui référence des tables qui n'ont pas été analysées.

Chaque fois qu'un ajout de données à une table non vide modifie de façon importante la taille de la table, vous pouvez mettre à jour explicitement les statistiques. Pour ce faire, vous pouvez exécuter une commande ANALYZE ou utiliser l'option STATUPDATE ON avec la commande COPY. Pour afficher les détails du nombre de lignes qui ont été insérées ou supprimées depuis la dernière opération ANALYZE, interrogez la table catalogue système PG_STATISTIC_INDICATOR.

Vous pouvez définir la portée de la commande ANALYSE en spécifiant l'une des valeurs suivantes :

- La totalité de la base de données actuelle
- Une seule table
- Une ou plusieurs colonnes spécifiques d'une seule table
- Colonnes qui sont susceptibles d'être utilisées comme prédicats dans des requêtes

La commande ANALYZE obtient un échantillon de lignes de la table, effectue quelques calculs et enregistre les statistiques de colonnes obtenues. Par défaut, Amazon Redshift exécute un exemple de passage pour la colonne DISTKEY et un autre exemple de passage pour toutes les autres colonnes de la table. Si vous souhaitez générer des statistiques pour un sous-ensemble de colonnes, vous pouvez spécifier une liste de colonnes séparées par des virgules. Vous pouvez exécuter

ANALYZE avec la clause PREDICATE COLUMNS pour ignorer les colonnes qui ne sont pas utilisées comme prédicats.

Comme les opérations ANALYZE sont gourmandes en ressources, exécutez-les uniquement sur des tables et des colonnes qui nécessitent des mises à jour des statistiques. Vous n'avez pas besoin d'analyser toutes les colonnes de toutes les tables régulièrement ou selon le même calendrier. Si les données changent considérablement, analysez les colonnes qui sont fréquemment utilisées dans les cas suivants :

- Tri et regroupement d'opérations
- Jointures
- Prédicats de requête

Pour réduire le délai de traitement et améliorer les performances globales du système, Amazon Redshift ignore ANALYZE pour chaque table dont le pourcentage de lignes modifiées est faible, comme déterminé par le paramètre analyze_threshold_percent. Par défaut, le seuil d'analyse est défini sur 10 %. Vous pouvez modifier le seuil d'analyse pour la séance en cours en exécutant une commande SET.

Les colonnes qui sont moins susceptibles d'exiger des analyses fréquentes sont celles qui représentent des faits et des mesures, ainsi que les attributs associés qui sont jamais réellement interrogés, comme les colonnes VARCHAR volumineuses. Par exemple, considérons la table LISTING de la base de données TICKIT.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'listing';


column         |        type        | encoding | distkey | sortkey
---------------+--------------------+----------+---------+---------
listid         | integer            | none     | t       | 1
sellerid       | integer            | none     | f       | 0
eventid        | integer            | mostly16 | f       | 0
dateid         | smallint           | none     | f       | 0
numtickets     | smallint           | mostly8  | f       | 0
priceperticket | numeric(8,2)       | bytedict | f       | 0
totalprice     | numeric(8,2)       | mostly32 | f       | 0
listtime       | timestamp with...  | none     | f       | 0
```

Si cette table est chargée tous les jours avec un grand nombre de nouveaux enregistrements, la colonne LISTID, qui est fréquemment utilisée dans les requêtes comme clé de jointure, doit être analysée régulièrement. Si TOTALPRICE et LISTTIME sont les contraintes fréquemment utilisées dans les requêtes, vous pouvez analyser ces colonnes et la clé de distribution sur chaque jour de la semaine.

```
analyze listing(listid, totalprice, listtime);
```

Supposons que les vendeurs et les événements de l'application soient beaucoup plus statiques et que la date IDs fasse référence à un ensemble de jours fixes couvrant seulement deux ou trois ans. Dans ce cas, les valeurs uniques de ces colonnes ne changent pas de façon significative. Cependant, le nombre d'instances de chaque valeur unique augmente de façon continue.

En outre, prenons l'exemple d'une situation où les mesures NUMTICKETS et PRICEPERTICKET sont interrogées peu fréquemment par rapport à la colonne TOTALPRICE. Dans ce cas, vous pouvez exécuter la commande ANALYZE sur l'ensemble de la table une fois tous les week-ends afin de mettre à jour les statistiques des cinq colonnes qui ne sont pas analysées quotidiennement :

Colonnes de prédicat

Comme solution pratique autre que spécifier une liste de colonnes, vous pouvez choisir d'analyser uniquement les colonnes susceptibles d'être utilisées comme prédicats. Quand vous exécutez une requête, les colonnes qui sont utilisées dans une jointure, une condition de filtre ou une clause group by sont marquées en tant que comme colonnes de prédicat dans le catalogue système. Lorsque vous exécutez ANALYZE avec la clause PREDICATE COLUMNS, l'opération d'analyse inclut uniquement les colonnes qui remplissent les critères suivants :

- La colonne est marquée en tant que colonne de prédicat.
- La colonne est une clé de distribution.
- La colonne fait partie d'une clé de tri.

Si aucune des colonnes d'une table n'est marquée comme prédicat, ANALYZE inclut toutes les colonnes, même si PREDICATE COLUMNS est spécifié. Si aucune colonne n'est marquée comme colonne de prédicat, c'est peut-être parce que la table n'a pas encore été interrogée.

Vous pouvez choisir d'utiliser PREDICATE COLUMNS lorsque votre modèle de requête de charge de travail est relativement stable. Lorsque le modèle de requête est variable, avec des colonnes

différentes souvent utilisées comme prédicats, l'utilisation de PREDICATE COLUMNS peut se traduire temporairement par des statistiques obsolètes. Des statistiques obsolètes peuvent donner lieu à des plans d'exécution de requête peu efficaces et à des lenteurs d'exécution. Cependant, la fois suivante où vous exécutez ANALYZE avec PREDICATE COLUMNS, les nouvelles colonnes de prédicat sont incluses.

Pour afficher des détails sur les colonnes de prédicat, utilisez la requête SQL suivante pour créer une vue nommée PREDICATE_COLUMNS.

```
CREATE VIEW predicate_columns AS
WITH predicate_column_info as (
SELECT ns.nspname AS schema_name, c.relname AS table_name, a.attnum as col_num,
 a.attname as col_name,
        CASE
            WHEN 10002 = s.stakind1 THEN array_to_string(stavalues1, '||')
            WHEN 10002 = s.stakind2 THEN array_to_string(stavalues2, '||')
            WHEN 10002 = s.stakind3 THEN array_to_string(stavalues3, '||')
            WHEN 10002 = s.stakind4 THEN array_to_string(stavalues4, '||')
            ELSE NULL::varchar
        END AS pred_ts
   FROM pg_statistic s
   JOIN pg_class c ON c.oid = s.starelid
   JOIN pg_namespace ns ON c.relnamespace = ns.oid
   JOIN pg_attribute a ON c.oid = a.attrelid AND a.attnum = s.staattnum)
SELECT schema_name, table_name, col_num, col_name,
       pred_ts NOT LIKE '2000-01-01%' AS is_predicate,
       CASE WHEN pred_ts NOT LIKE '2000-01-01%' THEN (split_part(pred_ts,
 '||',1))::timestamp ELSE NULL::timestamp END as first_predicate_use,
       CASE WHEN pred_ts NOT LIKE '%||2000-01-01%' THEN (split_part(pred_ts,
 '||',2))::timestamp ELSE NULL::timestamp END as last_analyze
FROM predicate_column_info;
```

Supposons que vous exécutez la requête suivante sur la table LISTING. Notez que les colonnes LISTID, LISTTIME et EVENTID sont utilisées dans les clauses de jointure, de filtre et group by.

```
select s.buyerid,l.eventid, sum(l.totalprice)
from listing l
join sales s on l.listid = s.listid
where l.listtime > '2008-12-01'
group by l.eventid, s.buyerid;
```

Lorsque vous interrogez la vue PREDICATE_COLUMNS, comme illustré dans l'exemple suivant, vous voyez que les colonnes LISTID, EVENTID et LISTTIME sont marquées en tant que colonnes de prédicat.

```
select * from predicate_columns
where table_name = 'listing';
```

```
schema_name | table_name | col_num | col_name       | is_predicate |
 first_predicate_use | last_analyze
------------+------------+---------+----------------+--------------
+---------------------+--------------------
public      | listing    |       1 | listid         | true         | 2017-05-05
 19:27:59 | 2017-05-03 18:27:41
public      | listing    |       2 | sellerid       | false        |
  | 2017-05-03 18:27:41
public      | listing    |       3 | eventid        | true         | 2017-05-16
 20:54:32 | 2017-05-03 18:27:41
public      | listing    |       4 | dateid         | false        |
  | 2017-05-03 18:27:41
public      | listing    |       5 | numtickets     | false        |
  | 2017-05-03 18:27:41
public      | listing    |       6 | priceperticket | false        |
  | 2017-05-03 18:27:41
public      | listing    |       7 | totalprice     | false        |
  | 2017-05-03 18:27:41
public      | listing    |       8 | listtime       | true         | 2017-05-16
 20:54:32 | 2017-05-03 18:27:41
```

Tenir les statistiques à jour améliore les performances des requêtes en permettant au planificateur de requête de choisir les plans optimaux. Amazon Redshift actualise automatiquement les statistiques en arrière-plan. Vous pouvez également exécuter explicitement la commande ANALYZE. Si vous choisissez d'exécuter explicitement ANALYZE, procédez comme suit :

- Exécutez la commande ANALYZE avant d'exécuter les requêtes.
- Exécutez la commande ANALYZE sur la base de données systématiquement à la fin de chaque chargement ou cycle de mise à jour régulier.
- Exécutez la commande ANALYZE sur les nouvelles tables que vous créez et sur les tables ou les colonnes existantes qui subissent des modifications significatives.

- Envisagez d'exécuter les opérations ANALYZE selon différents échéanciers pour différents types de tables et de colonnes, en fonction de leur utilisation dans les requêtes et leur propension au changement.
- Pour gagner du temps et économiser des ressources de cluster, utilisez la clause PREDICATE COLUMNS lorsque vous exécutez ANALYZE.

Vous n'avez pas besoin d'exécuter explicitement la commande ANALYZE après la restauration d'un instantané sur un cluster provisionné ou un espace de noms sans serveur, ni après la reprise d'un cluster provisionné suspendu. En pareils cas, Amazon Redshift conserve les informations de table système, ce qui rend les commandes ANALYZE manuelles inutiles. Amazon Redshift continuera à exécuter des opérations d'analyse automatique selon les besoins.

Une opération d'analyse ignore les tables contenant des up-to-date statistiques. Si vous exécutez ANALYZE dans le cadre de votre flux de travail ETL (extraction, transformation et chargement), l'analyse automatique ignore les tables dont les statistiques sont à jour. De façon similaire, une opération ANALYZE explicite ignore les tables pour lesquelles l'analyse automatique a mis à jour les statistiques.

Historique de la commande ANALYZE

Il est utile de savoir quand la commande ANALYZE a été exécutée pour la dernière fois sur une table ou une base de données. Quand une commande ANALYZE est exécutée, Amazon Redshift exécute plusieurs requêtes qui ressemblent à ceci :

```
padb_fetch_sample: select * from table_name
```

Interrogez STL_ANALYZE pour afficher l'historique des opérations d'analyse. Si Amazon Redshift analyse une table à l'aide d'une analyse automatique, la colonne `is_background` est définie sur `t` (true). Sinon, elle est définie sur `f` (false). L'exemple suivant joint la table STV_TBL_PERM pour afficher le nom de la table et les détails de l'exécution.

```
select distinct a.xid, trim(t.name) as name, a.status, a.rows, a.modified_rows,
 a.starttime, a.endtime
from stl_analyze a
join stv_tbl_perm t  on t.id=a.table_id
where name = 'users'
order by starttime;
```

```
xid    | name  | status          | rows  | modified_rows | starttime           |
 endtime
-------+-------+-----------------+-------+---------------+---------------------
+---------------------
  1582 | users | Full            | 49990 |         49990 | 2016-09-22 22:02:23 |
 2016-09-22 22:02:28
244287 | users | Full            | 24992 |         74988 | 2016-10-04 22:50:58 |
 2016-10-04 22:51:01
244712 | users | Full            | 49984 |         24992 | 2016-10-04 22:56:07 |
 2016-10-04 22:56:07
245071 | users | Skipped         | 49984 |             0 | 2016-10-04 22:58:17 |
 2016-10-04 22:58:17
245439 | users | Skipped         | 49984 |          1982 | 2016-10-04 23:00:13 |
 2016-10-04 23:00:13
(5 rows)
```

Sinon, vous pouvez exécuter une requête plus complexe qui renvoie toutes les instructions exécutées dans chaque transaction terminée et qui incluaient une commande ANALYZE :

```
select xid, to_char(starttime, 'HH24:MM:SS.MS') as starttime,
datediff(sec,starttime,endtime ) as secs, substring(text, 1, 40)
from svl_statementtext
where sequence = 0
and xid in (select xid from svl_statementtext s where s.text like 'padb_fetch_sample
%' )
order by xid desc, starttime;

xid  |  starttime   | secs |                 substring
-----+--------------+------+-------------------------------------------
1338 | 12:04:28.511 |    4 | Analyze date
1338 | 12:04:28.511 |    1 | padb_fetch_sample: select count(*) from
1338 | 12:04:29.443 |    2 | padb_fetch_sample: select * from date
1338 | 12:04:31.456 |    1 | padb_fetch_sample: select * from date
1337 | 12:04:24.388 |    1 | padb_fetch_sample: select count(*) from
1337 | 12:04:24.388 |    4 | Analyze sales
1337 | 12:04:25.322 |    2 | padb_fetch_sample: select * from sales
1337 | 12:04:27.363 |    1 | padb_fetch_sample: select * from sales
...
```

Exécution de l'opération VACUUM sur les tables

Amazon Redshift peut trier et effectuer automatiquement une opération VACUUM DELETE sur les tables en arrière-plan. Pour nettoyer les tables après un chargement ou une série de mises à jour incrémentielles, vous pouvez également exécuter la commande VACUUM, sur toute la base de données ou sur chaque table individuelle.

> **Note**
>
> Seuls les utilisateurs disposant des autorisations nécessaires peuvent efficacement vider une table. Si VACUUM est exécutée sans les autorisations de table nécessaires, l'opération se termine correctement, mais n'a aucun effet. Pour obtenir la liste des autorisations de table valides permettant d'exécuter VACUUM efficacement, consultez VACUUM.
> C'est pourquoi nous vous recommandons de nettoyer les tables individuellement en fonction des besoins. Nous vous recommandons également cette approche, car le nettoyage de toute la base de données peut être une opération coûteuse.

Tri automatique des tables

Amazon Redshift trie automatiquement les données en arrière-plan pour gérer les données de la table selon l'ordre de la clé de tri. Amazon Redshift assure le suivi de vos requêtes d'analyse afin de déterminer à quelles sections de la table le tri sera appliqué.

Selon la charge du système, Amazon Redshift lance le tri automatiquement. Ce tri automatique réduit la nécessité d'exécuter la commande VACUUM pour que les données restent dans l'ordre de la clé de tri. Si vous souhaitez que les données soient triées dans l'ordre de la clé de tri, par exemple après un chargement de données volumineux, vous pouvez toujours exécuter la commande VACUUM manuellement. Afin de déterminer s'il sera utile pour votre table d'exécuter une commande VACUUM SORT, surveillez la colonne `vacuum_sort_benefit` dans SVV_TABLE_INFO.

Amazon Redshift suit les requêtes d'analyse qui utilisent la clé de tri de chaque table. Amazon Redshift estime le pourcentage maximum d'amélioration en analysant et filtrant les données de chaque table (si la table a été complètement triée). L'estimation est visible dans la colonne `vacuum_sort_benefit` de SVV_TABLE_INFO. Vous pouvez utiliser cette colonne avec la colonne `unsorted` afin de déterminer à quel moment il est utile pour les requêtes d'exécuter une opération VACUUM SORT sur une table. La colonne `unsorted` reflète l'ordre de tri physique d'une table. La

colonne `vacuum_sort_benefit` spécifie l'impact du tri d'une table en exécutant manuellement une opération VACUUM SORT.

Par exemple, réfléchissez à la requête suivante :

```
select "table", unsorted,vacuum_sort_benefit from svv_table_info order by 1;
```

```
 table | unsorted | vacuum_sort_benefit
-------+----------+---------------------
 sales |    85.71 |                5.00
 event |    45.24 |               67.00
```

Pour la table « sales », même si la table n'est pas triée à ~86 %, l'impact de la requête sur la performance pour la table est de 5 % seulement. Cela peut être parce qu'une petite partie de la table a été utilisée par les requêtes ou parce qu'un petit nombre de requêtes ont eu accès à la table. Pour la table « event », elle n'est pas triée physiquement à ~45 %. Mais l'impact de la requête sur la performance est de 67 %, ce qui indique qu'une partie plus importante de la table a été utilisée pour les requêtes ou qu'un grand nombre de requêtes ont eu accès à la table. L'exécution de l'opération VACUUM SORT peut potentiellement être utile pour la table « event ».

Suppression de vide automatique

Lorsque vous effectuez une suppression, les lignes sont marquées pour suppression, mais ne sont pas supprimées. Amazon Redshift exécute automatiquement une opération VACUUM DELETE en arrière-plan en fonction du nombre de lignes supprimées dans les tables de base de données. Amazon Redshift planifie l'exécution de VACUUM DELETE au cours des périodes à charge réduite et suspend l'opération au cours des périodes à charge élevée.

Rubriques

- Fréquence de VACUUM
- Phase de tri et phase de fusion
- Seuil de VACUUM
- Types d'opération VACUUM
- Minimiser les temps d'aspiration

Fréquence de VACUUM

Vous devez exécuter la commande VACUUM aussi souvent que nécessaire pour assurer des performances de requêtes constantes. Prenez en compte ces facteurs au moment de déterminer la fréquence d'exécution de votre commande VACUUM :

- Exécutez la commande VACUUM pendant les périodes où vous prévoyez une activité minimale sur le cluster, telles que le soir ou les fenêtres dédiées d'administration de la base de données.
- Exécutez les commandes VACUUM en dehors des fenêtres de maintenance. Pour plus d'informations, consultez Planifier la maintenance Windows.
- Une grande région non triée se traduit par des temps d'exécution de la commande VACUUM plus longs. Si vous différez la commande, son exécution prendra plus de temps, car un plus grand nombre de données doit être réorganisé.
- VACUUM est une opération I/O intensive. Par conséquent, plus la réalisation de votre aspirateur est longue, plus elle aura d'impact sur les requêtes simultanées et les autres opérations de base de données exécutées sur votre cluster.
- VACUUM prend plus de temps pour les tables qui utilisent le tri entrelacé. Pour déterminer si les tables entrelacées doivent être retriées, interrogez la vue SVV_INTERLEAVED_COLUMNS.

Phase de tri et phase de fusion

Amazon Redshift effectue une opération VACUUM en deux étapes : tout d'abord, il trie les lignes de la région non triée, puis, si nécessaire, il fusionne les lignes nouvellement triées en fin de la table avec les lignes existantes. Lors de l'exécution de la commande VACUUM sur une grande table, l'opération procède par une série d'étapes consistant en tris incrémentiels suivis de fusions. Si l'opération échoue ou si Amazon Redshift passe hors connexion pendant l'opération, la table ou la base de données partiellement aspirées se trouvent dans un état cohérent, mais vous devez relancer manuellement l'opération VACUUM. Les tris incrémentiels sont perdus, mais les lignes fusionnées qui ont été validées avant la défaillance n'ont pas besoin d'être aspirées à nouveau. Si la région non triée est grande, le temps perdu peut être important. Pour plus d'informations sur les étapes de tri et de fusion, consultez Réduire le volume des lignes fusionnées.

Les utilisateurs peuvent accéder aux tables pendant qu'elles sont aspirées. Vous pouvez exécuter des requêtes et des opérations d'écriture pendant qu'une table est l'objet de la commande VACUUM, mais lorsque DML et une commande VACUUM s'exécutent en même temps, les deux peuvent prendre plus de temps. Si vous exécutez les instructions UPDATE et DELETE pendant une opération

VACUUM, les performances du système peuvent être réduites. Les fusions incrémentielles bloquent temporairement les opérations UPDATE et DELETE simultanée, tandis que les opérations UPDATE et DELETE bloquent à leur tour temporairement les étapes de la fusion incrémentielle sur les tables concernées. Les opérations DDL, telles que ALTER TABLE, sont bloquées tant que l'opération VACUUM sur la table n'est pas finie.

> **Note**
>
> Divers modificateurs pour l'opération VACUUM contrôlent la façon dont elle fonctionne. Vous pouvez les utiliser pour adapter l'opération aux besoins actuels. Par exemple, l'utilisation de VACUUM RECLUSTER accélère l'opération VACUUM, car elle n'effectue pas d'opération de fusion complète. Pour plus d'informations, consultez VACUUM.

Seuil de VACUUM

Par défaut, la commande VACUUM ignore la phase de tri pour toute table dans laquelle plus de 95 % des lignes sont déjà triées. L'omission de la phase de tri peut améliorer considérablement les performances de l'opération VACUUM. Pour modifier le seuil de tri par défaut d'une seule table, incluez le nom de la table et le paramètre TO seuil PERCENT lorsque vous exécutez la commande VACUUM.

Types d'opération VACUUM

Pour plus d'informations sur les différents types d'opérations VACUUM, consultez VACUUM.

Minimiser les temps d'aspiration

Amazon Redshift trie automatiquement les données et exécute l'opération VACUUM DELETE en arrière-plan. Cela réduit la nécessité d'exécuter la commande VACUUM. Passer l'aspirateur est un processus qui peut prendre beaucoup de temps. En fonction de la nature de vos données, nous recommandons les pratiques suivantes afin de minimiser les temps d'aspiration.

Rubriques

- Décidez s'il faut réindexer
- Réduire la taille de la région non triée
- Réduire le volume des lignes fusionnées

- Chargez vos données dans l'ordre des clés de tri
- Utilisez des tables de séries chronologiques pour réduire les données stockées

Décidez s'il faut réindexer

Vous pouvez souvent améliorer de façon significative les performances des requêtes en utilisant un style de tri entrelacé, mais au fil du temps les performances peuvent se dégrader si la distribution des valeurs des colonnes de clé de tri change.

Lorsque vous chargez initialement une table entrelacée vide à l'aide de COPY ou CREATE TABLE AS, Amazon Redshift crée automatiquement l'index entrelacé. Si vous chargez initialement une table entrelacée à l'aide d'INSERT, vous devez exécuter VACUUM REINDEX après pour initialiser l'index entrelacé.

Au fil du temps, à mesure que vous ajoutez des lignes avec de nouvelles valeurs de clé de tri, les performances peuvent se dégrader si la distribution des valeurs dans les colonnes de clés change. Si vos nouvelles lignes se trouvent principalement dans la plage de valeurs des clés de tri existantes, vous n'avez pas besoin de réindexer. Exécutez VACUUM SORT ONLY ou VACUUM FULL pour rétablir l'ordre de tri.

Le moteur de requête est en mesure d'utiliser l'ordre de tri pour sélectionner efficacement les blocs de données qui doivent être analysés pour traiter une requête. Pour un tri entrelacé, Amazon Redshift analyse les valeurs des colonnes de clé de tri pour déterminer l'ordre de tri optimal. Si la distribution des valeurs de clés change, ou est altérée, au fur et à mesure que les lignes sont ajoutées, la politique de tri n'est plus optimale et l'avantage des performances de tri se dégrade. Pour réanalyser la distribution des clés de tri, vous pouvez exécuter une opération VACUUM REINDEX. Comme l'opération de réindexation prend du temps, pour décider si une table peut bénéficier d'une réindexation, interrogez la vue SVV_INTERLEAVED_COLUMNS.

Par exemple, la requête suivante affiche les détails des tables qui utilisent les clés de tri entrelacé.

```
select tbl as tbl_id, stv_tbl_perm.name as table_name,
col, interleaved_skew, last_reindex
from svv_interleaved_columns, stv_tbl_perm
where svv_interleaved_columns.tbl = stv_tbl_perm.id
and interleaved_skew is not null;


 tbl_id | table_name | col | interleaved_skew | last_reindex
--------+------------+-----+------------------+--------------------
```

```
 100048 | customer   |   0 |            3.65 | 2015-04-22 22:05:45
 100068 | lineorder  |   1 |            2.65 | 2015-04-22 22:05:45
 100072 | part       |   0 |            1.65 | 2015-04-22 22:05:45
 100077 | supplier   |   1 |            1.00 | 2015-04-22 22:05:45
(4 rows)
```

La valeur de `interleaved_skew` est un rapport qui indique le degré de déformation. Une valeur égale à 1 signifie qu'il n'y a pas de déformation. Si la déformation est supérieure à 1,4, une opération VACUUM REINDEX améliore généralement les performances, sauf si la déformation est inhérente à l'ensemble jeu sous-jacent.

Vous pouvez utiliser la valeur de date dans `last_reindex` pour déterminer le temps qui s'est écoulé depuis la dernière réindexation.

Réduire la taille de la région non triée

La région non triée croît lors du chargement de grandes quantités de nouvelles données dans des tables qui contiennent déjà des données ou lorsque vous ne pas videz pas les tables dans le cadre de vos opérations régulières de maintenance. Pour éviter les longues opérations VACUUM, utilisez les pratiques suivantes :

- Exécutez les opérations VACUUM sur une base régulière.

  Si vous chargez vos tables par petits incréments (mises à jour quotidiennes qui représentent un faible pourcentage du nombre total de lignes de la table, par exemple), l'exécution régulière de VACUUM aide à s'assurer que les opérations VACUUM individuelles se déroulent rapidement.
- Exécutez d'abord le chargement le plus important.

  Si vous avez besoin de charger une nouvelle table avec plusieurs opérations COPY, exécutez d'abord le chargement le plus important. Lorsque vous exécutez un chargement initial dans une table nouvelle ou tronquée, toutes les données sont chargées directement dans la région triée et, par conséquent, aucune opération VACUUM n'est obligatoire.
- Tronquez une table au lieu de supprimer toutes les lignes.

  La suppression des lignes d'une table ne récupère pas l'espace que les lignes occupaient jusqu'à ce que vous effectuiez une opération VACUUM ; cependant, la troncation d'une table vide la table et récupère l'espace, et, par conséquent, aucune opération VACUUM n'est obligatoire. Une autre solution consiste à supprimer la table et à la recréer.
- Tronquez ou supprimez les tables de test.

Si vous chargez un petit nombre de lignes dans une table à des fins de test, ne supprimez pas les lignes lorsque vous avez terminé. A la place, tronquez la table et rechargez les lignes dans le cadre de l'opération de chargement de production suivante.

- Exécutez une copie complète.

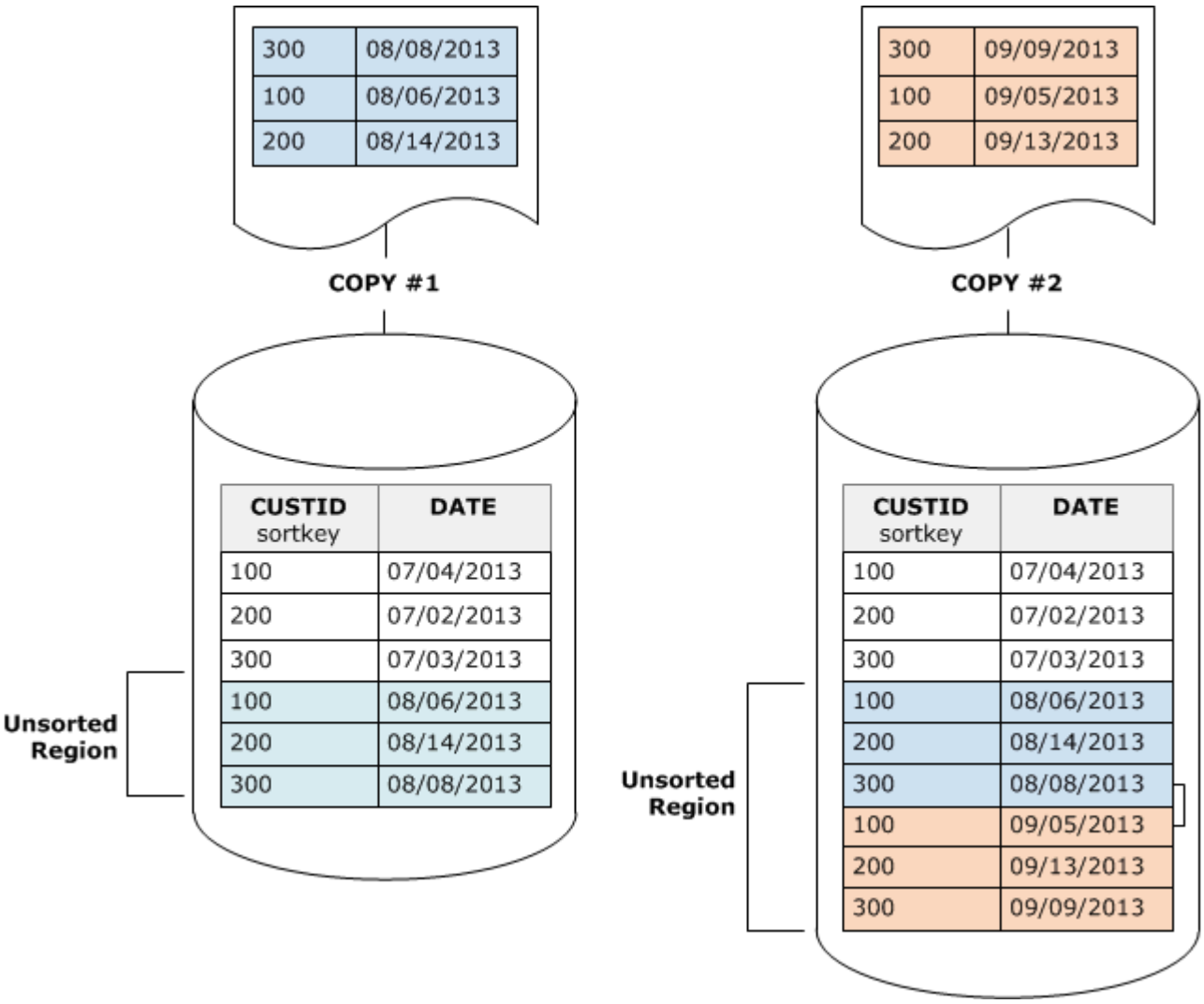
  Si une table qui utilise une table de clé de tri composée possède une grande région non triée, une copie complète est beaucoup plus rapide qu'une opération VACUUM. Une copie complète recrée et remplit une table à l'aide d'une insertion en bloc, qui retrie automatiquement la table. Si une table possède une grande région non triée, une copie complète est beaucoup plus rapide qu'une opération VACUUM. Cependant, vous ne pouvez pas effectuer de mises à jour simultanées pendant une opération de copie complète, alors que cela est possible durant une opération VACUUM. Pour de plus amples informations, veuillez consulter Bonnes pratiques Amazon Redshift pour la conception de requêtes.

Réduire le volume des lignes fusionnées

Si une opération VACUUM doit fusionner de nouvelles lignes dans la région triée d'une table, le temps nécessaire pour l'opération augmente au fur et à mesure que la table se développe. Vous pouvez améliorer les performances de l'opération VACUUM en réduisant le nombre de lignes à fusionner.

Avant une opération VACUUM, une table se compose d'une région triée en tête de table, suivie d'une région non triée, qui croît chaque fois que des lignes sont ajoutées ou mises à jour. Lorsqu'un ensemble de lignes est ajouté par une opération COPY, le nouvel ensemble de lignes est trié sur la clé de tri tel qu'il est ajouté à la région non triée en fin de table. Les nouvelles lignes sont classées au sein de leur propre ensemble, mais pas au sein de la région non triée.

Le schéma suivant illustre la région non triée après deux opérations COPY successives, où la clé de tri est CUSTID. Pour plus de simplicité, cet exemple montre une clé de tri composée, mais les mêmes principes s'appliquent aux clés de tri entrelacé, sauf que l'impact de la région non triée est une plus grande pour les tables entrelacées.



Une opération VACUUM restaure l'ordre de tri de la table en deux étapes :

1. Triez la région non triée dans une région nouvellement triée.

   La première étape est relativement bon marché, parce que seule la région non triée est réécrite. Si la plage des valeurs de clé de tri de la région nouvellement triée est supérieure à la plage existante, seules les nouvelles lignes doivent être réécrites, et l'opération VACUUM est terminée. Par exemple, si la région triée contient des valeurs d'ID comprises entre 1 et 500 et que les opérations de copie suivantes ajoutent des valeurs de clé supérieures à 500, seule la région non triée doit être réécrite.

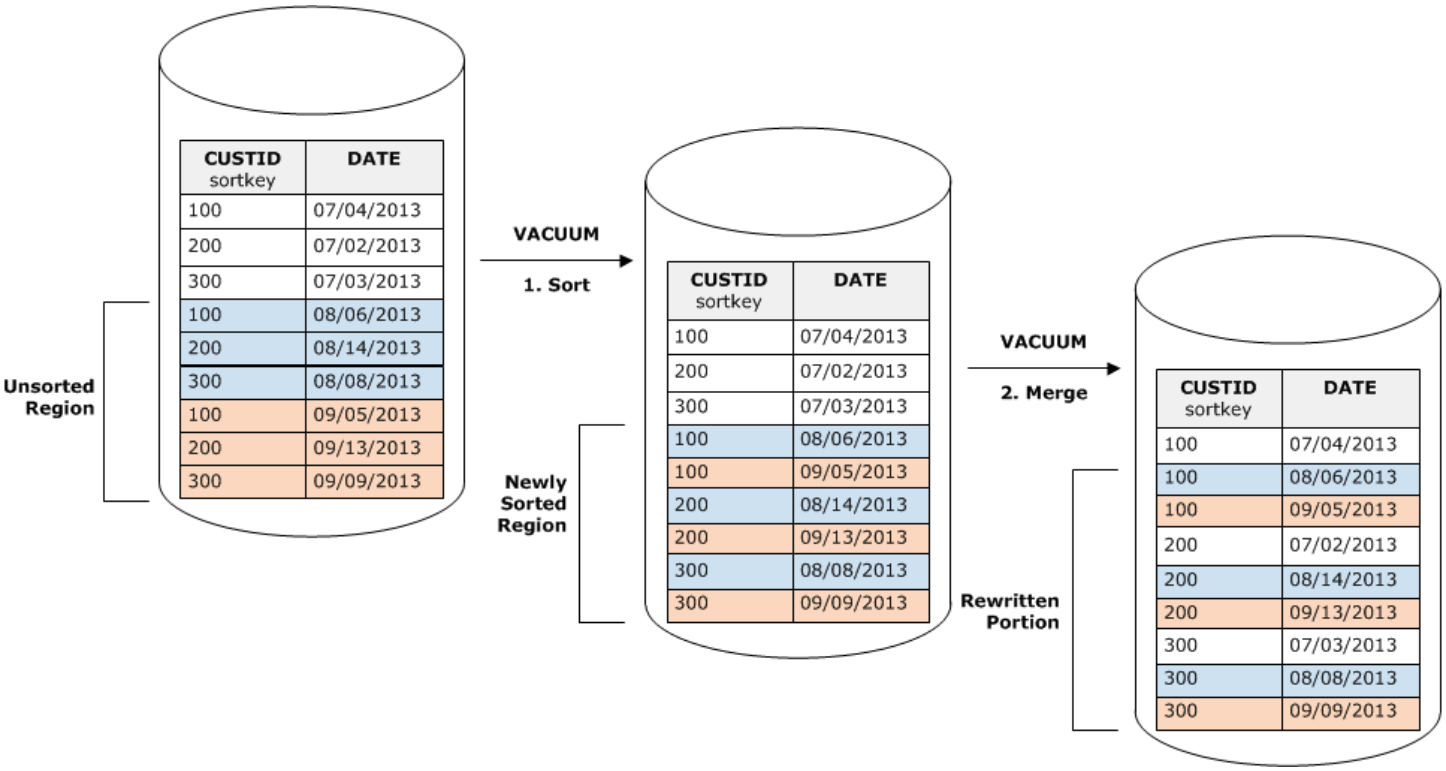
2. Fusionnez la région nouvellement triée avec la région précédemment triée.

   Si les clés de la région nouvellement triée chevauchent les clés de la région triée, l'opération VACUUM doit fusionner les lignes. En commençant par le début de la région nouvellement triée

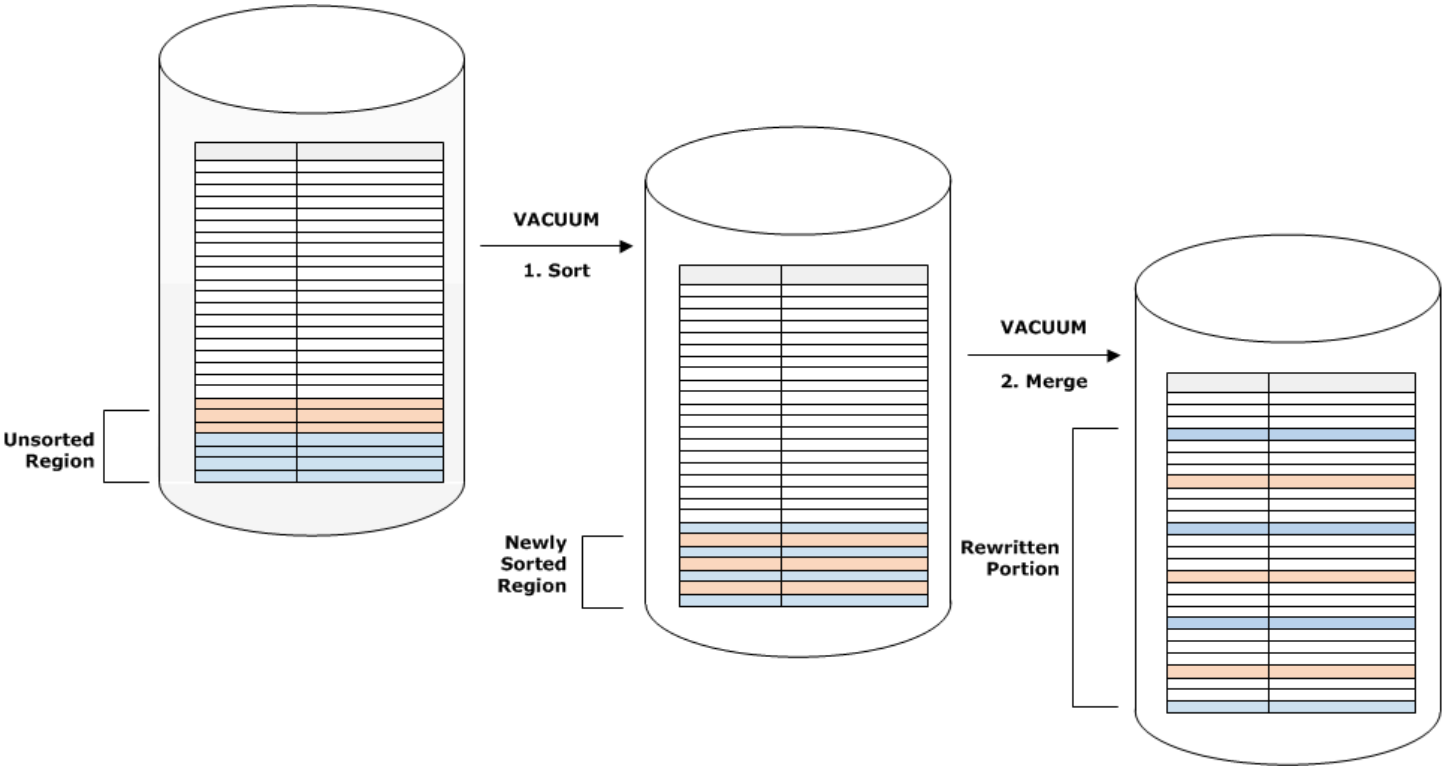
(à la clé de tri la plus basse), l'opération VACUUM écrit les lignes fusionnées à partir de la région précédemment triée et de la région nouvellement triée dans un nouvel ensemble de blocs.

L'étendue selon laquelle la nouvelle plage de clés de tri chevauche les clés de tri existantes détermine l'étendue selon laquelle la région précédemment triée doit être réécrite. Si les clés non triées sont dispersées à travers la plage de tri existante, une opération VACUUM peut avoir besoin de réécrire Des parties existantes de la table.

Le schéma suivant montre comment une opération VACUUM trie et fusionne les lignes qui sont ajoutées à une table où CUSTID est la clé de tri. Comme chaque opération de copie ajoute un nouvel ensemble de lignes avec des valeurs de clé qui chevauchent les clés existantes, presque la totalité de la table doit être réécrite. Le schéma illustre une seule étape de tri et fusion, mais, en pratique, une grand opération VACUUM se compose d'une série d'étapes incrémentielles de tri et de fusion.



Si la plage de clés de tri d'un ensemble de nouvelles lignes chevauche la plage des clés existantes, le coût de l'étape de fusion continue à croître proportionnellement à la taille de la table au fur et à mesure que la table augmente, tandis que le coût de l'étape de tri demeure proportionnel à la taille de la région non triée. Dans un tel cas, le coût de l'étape de fusion éclipse le coût de l'étape de tri, comme l'illustre le schéma suivant.



Pour déterminer quelle proportion d'une table a été refusionnée, interrogez SVV_VACUUM_SUMMARY après la fin de l'opération VACUUM. La requête suivante affiche les conséquences de six opérations VACUUM successives tandis que CUSTSALES croît au fil du temps.

```
select * from svv_vacuum_summary
where table_name = 'custsales';


 table_name | xid  | sort_      | merge_     | elapsed_   | row_  | sortedrow_ | block_
  | max_merge_
            |      | partitions | increments | time       | delta | delta      | delta
  | partitions
 -----------+------+------------+------------+------------+-------+------------
+---------+---------------
  custsales | 7072 |          3 |          2 |  143918314 |     0 |   88297472 |   1524
  |      47
  custsales | 7122 |          3 |          3 |  164157882 |     0 |   88297472 |    772
  |      47
  custsales | 7212 |          3 |          4 |  187433171 |     0 |   88297472 |    767
  |      47
  custsales | 7289 |          3 |          4 |  255482945 |     0 |   88297472 |    770
  |      47
```

```
 custsales | 7420 |          3 |          5 |  316583833 |     0 |   88297472 |    769
 |      47
 custsales | 9007 |          3 |          6 |  306685472 |     0 |   88297472 |    772
 |      47
(6 rows)
```

La colonne merge_increments fournit une indication de la quantité de données qui a été fusionnée pour chaque opération VACUUM. Si le nombre d'incréments de fusion sur des opérations VACUUM consécutives augmente proportionnellement à la croissance de la taille de la table, cela indique que chaque opération VACUUM refusionne un nombre croissant de lignes de la table, car la région existante et la région nouvellement triée se chevauchent.

Chargez vos données dans l'ordre des clés de tri

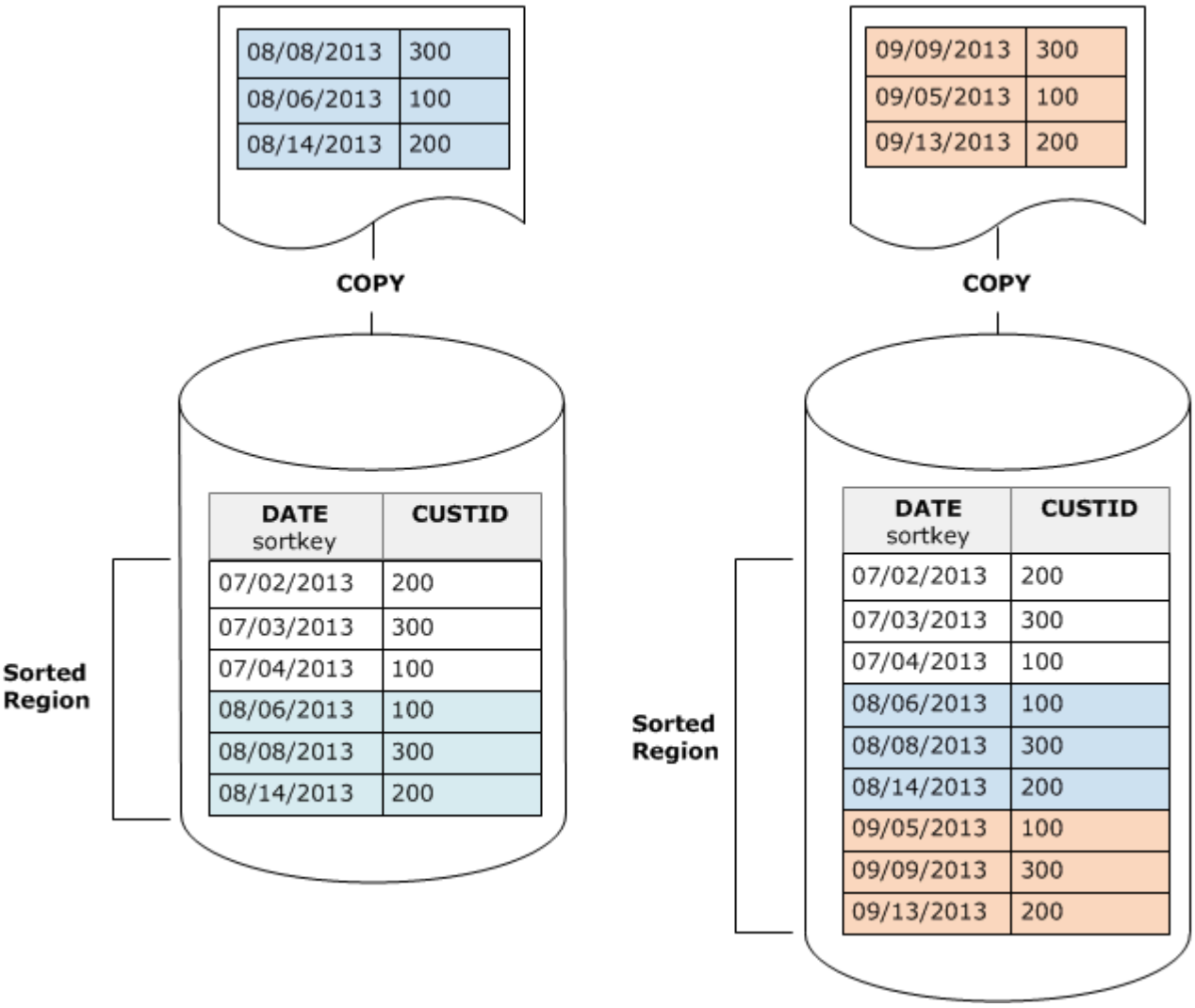
Si vous chargez vos données dans l'ordre de la clé de tri avec une commande COPY, vous aurez peut-être moins (voire plus du tout) besoin d'avoir recours à l'opération VACUUM.

La commande COPY ajoute automatiquement de nouvelles lignes à la région triée de la table lorsque toutes les conditions suivantes sont définies sur true :

- La table utilise une clé de tri composée avec une seule colonne de tri.
- La colonne de tri est NOT NULL.
- La table est triée ou vide à 100 %.
- Toutes les nouvelles lignes sont plus élevées dans l'ordre de tri que les lignes existantes, y compris les lignes marquées pour la suppression. Dans ce cas, Amazon Redshift utilise les huit premiers octets de la clé de tri pour déterminer l'ordre de tri.

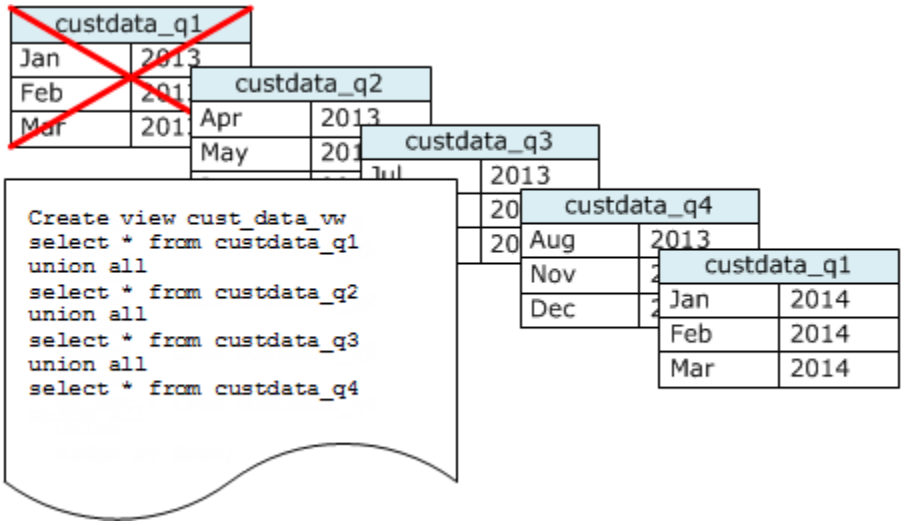
Par exemple, supposons que vous ayez une table qui enregistre les événements clients à l'aide d'un ID client et de l'heure. Si vous triez sur l'ID client, il est probable que la plage des clés de tri des nouvelles lignes ajoutées par les chargements incrémentiels chevauchent la plage existante, comme illustré dans l'exemple précédent, ce qui conduit à une opération VACUUM coûteuse.

Si vous définissez votre clé de tri sur une colonne d'horodatage, vos nouvelles lignes sont ajoutées dans l'ordre de tri à la fin de la table, comme l'illustre le schéma suivant, ce qui rend l'opération VACUUM moins (voire plus du tout) nécessaire.



Utilisez des tables de séries chronologiques pour réduire les données stockées

Si vous maintenez les données pendant une période aléatoire, utilisez une série de tables, comme l'illustre le schéma suivant.



Créez une table chaque fois que vous ajoutez un ensemble de données, puis supprimez la table la plus ancienne de la série. Vous bénéficiez d'un double avantage :

- Vous évitez le coût supplémentaire de suppression des lignes, parce qu'une opération DROP TABLE est beaucoup plus efficace qu'une opération DELETE massive.
- Si les tables sont triées par horodate, aucune opération VACUUM n'est nécessaire. Si chaque table contient les données pour un mois, une opération VACUUM devra au plus réécrire la valeur d'un mois de données, même si les tables ne sont pas triées par horodatage.

Vous pouvez créer une vue UNION ALL à utiliser par les requêtes de création de rapports qui masque le fait que les données sont stockées en plusieurs tables. Si une requête filtre sur la clé de tri, le planificateur de requête peut efficacement ignorer toutes les tables qui ne sont pas utilisées. Comme une opération UNION ALL peut être moins efficace pour les autres types de requêtes, vous devez évaluer les performances de la requête dans le contexte de toutes les requêtes qui utilisent les tables.

Gestion des opérations d'écriture simultanées

Amazon Redshift permet que les tables soient lues tandis qu'elles sont modifiées ou chargées de façon incrémentielle.

Dans certaines applications traditionnelles d'entrepôt des données et de Business Intelligence, la base de données est disponible pour les utilisateurs uniquement lorsque la charge nocturne est terminée. Dans ce cas, aucune mise à jour n'est autorisée pendant les heures de travail normales, lorsque les requêtes analytiques sont exécutées et les rapports générés ; cependant, un nombre

croissant d'applications demeurent actives pendant de longues périodes de la journée, voire toute la journée, ce qui rend obsolète l'idée d'une fenêtre de chargement.

Amazon Redshift prend en charge ces types d'applications en permettant que les tables soient lues tandis qu'elles sont chargées ou modifiées de façon incrémentielle. Les requêtes voient simplement la dernière version validée, ou instantané, des données plutôt que d'attendre la validation de la version suivante. Si vous souhaitez qu'une requête donnée attende la validation d'une autre opération d'écriture, vous devez le planifier en conséquence.

Les rubriques suivantes décrivent certains des concepts et scénarios clés qui impliquent les transactions, les instantanés de base de données, les mises à jour et le comportement simultané.

Rubriques

- Isolement sérialisable
- Écriture et read/write opérations
- Exemples d'écritures simultanées

Isolement sérialisable

Certaines applications nécessitent non seulement une interrogation et un chargement simultanés, mais également la possibilité d'écrire sur plusieurs tables ou sur la même table simultanément. Dans ce contexte, simultanément entraîne un chevauchement, non planifié pour s'exécuter exactement au même moment. Deux transactions sont considérées comme simultanées si la seconde démarre avant que la première ne soit validée. Les opérations simultanées peuvent provenir de différentes séances contrôlées par le même utilisateur ou par différents utilisateurs.

> **Note**
>
> Amazon Redshift prend en charge un comportement de validation automatique par défaut dans lequel chaque commande SQL exécutée séparément est validée individuellement. Si vous placez un ensemble de commandes dans un bloc de transaction (défini par les instructions BEGIN et FIN), le bloc est validé comme une seule transaction et, par conséquent, vous pouvez le restaurer si nécessaire. Les exceptions à ce comportement sont les commandes TRUNCATE et VACUUM, qui valident automatiquement toutes les modifications en attente effectuées dans la transaction actuelle.
> Certains clients SQL émettent automatiquement des commandes BEGIN et COMMIT, de sorte que le client contrôle si un groupe d'instructions est exécuté en tant que transaction

ou si chacune des instructions est exécutée indépendamment. Consultez la documentation relative à de l'interface que vous utilisez. Par exemple, lors de l'utilisation du pilote JDBC Amazon Redshift, un `PreparedStatement` JDBC avec une chaîne de requête qui contient plusieurs commandes SQL (séparées par des points-virgules) exécute toutes les instructions en tant qu'une seule transaction. En revanche, si vous utilisez SQL Workbench/J et que vous définissez AUTO COMMIT ON, si vous exécutez plusieurs instructions, chaque instruction s'exécute comme une transaction distincte.

Les opérations d'écriture simultanées sont prises en charge dans Amazon Redshift de manière protective, à l'aide de verrous d'écriture sur les tables et du principe d'isolement sérialisable. L'isolement sérialisable préserve l'illusion qu'une transaction s'exécutant sur une table est la seule transaction qui soit en cours d'exécution sur cette table. Par exemple, les deux transactions T1 et T2 s'exécutant simultanément doivent produire les mêmes résultats comme conséquence de l'une des conditions suivantes :

- T1 et T2 sont exécutées en série dans cet ordre.
- T2 et T1 sont exécutées en série dans cet ordre.

Les transactions simultanées sont invisibles les unes aux autres ; elles ne peuvent pas détecter les modifications de chacune d'elles. Chaque transaction simultanée crée un instantané de la base de données au début de la transaction. Un instantané de base de données est créé au sein d'une transaction sur la première occurrence de la plupart des instructions SELECT, des commandes DML telles que COPY, DELETE, INSERT, UPDATE et TRUNCATE, et des commandes DDL suivantes :

- ALTER TABLE (pour ajouter ou supprimer des colonnes)
- CREATE TABLE
- DROP TABLE
- TRUNCATE TABLE

Si une exécution en série des transactions simultanées produit les mêmes résultats que leur exécution simultanée, ces transactions sont considérées comme « sérialisables » et peuvent être exécutées en toute sécurité. Si aucune exécution en série de ces transactions ne peut produire les mêmes résultats, la transaction qui exécute une instruction qui pourrait interrompre la mise en série est arrêtée et annulée.

Les tables catalogue système (PG) et autres tables système Amazon Redshift (STL et STV) ne sont pas verrouillées dans une transaction. Par conséquent, les modifications apportées aux objets de base de données qui proviennent d'opérations DDL et TRUNCATE sont visibles lors de la validation de transactions simultanées.

Par exemple, supposons que cette table A existe dans la base de données lorsque deux transactions simultanées, T1 et T2, démarrent. Supposons que T2 renvoie une liste de tables en les sélectionnant dans la table de catalogue PG_TABLES. Ensuite, T1 supprime la table A et valide la suppression, puis T2 affiche les tables à nouveau. La table A n'apparaît plus dans la liste. Si T2 essaie d'interroger la table supprimée, Amazon Redshift renvoie une erreur indiquant que la relation n'existe pas. La requête de catalogue qui renvoie la liste des tables à T2 ou vérifie que la table A existe n'est pas soumise aux mêmes règles d'isolement que les opérations réalisées sur les tables de l'utilisateur.

Les transactions pour les mises à jour de ces tables s'exécutent dans un mode d'isolement validé en lecture. Les tables de catalogue préfixées par PG ne prennent pas en charge l'isolement d'instantané.

Isolement sérialisable pour les tables système et les tables de catalogue

Un instantané de base de données est également créé dans une transaction pour toute requête SELECT qui fait référence à une table créée par l'utilisateur ou à une table système Amazon Redshift (STL ou STV). Les requêtes SELECT qui ne font référence à aucune table ne créent pas d'instantané de base de données de transaction. Les instructions INSERT, DELETE et UPDATE qui fonctionnent uniquement sur les tables catalogue système (PG) ne créent pas non plus d'instantané de base de données de transaction.

Comment corriger les erreurs d'isolement sérialisable

ERROR:1023 DETAIL : Violation d'isolement sérialisable sur une table dans Redshift

Lorsqu'Amazon Redshift détecte une erreur d'isolement sérialisable, le message d'erreur suivant s'affiche.

```
ERROR:1023 DETAIL: Serializable isolation violation on table in Redshift
```

Pour corriger une erreur d'isolement sérialisable, vous pouvez essayer l'une des méthodes suivantes :

- Réessayez la transaction annulée.

Amazon Redshift a détecté qu'une charge de travail simultanée n'est pas sérialisable. Il suggère des lacunes dans la logique de l'application, qui peuvent généralement être contournées en réessayant la transaction qui a rencontré l'erreur. Si le problème persiste, essayez l'une des autres méthodes.

- Déplacez les opérations qui ne doivent pas se trouver dans la même transaction atomique hors de la transaction.

  Cette méthode s'applique lorsque des opérations individuelles à l'intérieur de deux transactions se comparent entre elles d'une façon pouvant affecter le résultat de l'autre transaction. Par exemple, les deux séances suivantes lancent chacune une transaction.

```
Session1_Redshift=# begin;
```

```
Session2_Redshift=# begin;
```

  Le résultat d'une instruction SELECT dans chaque transaction peut être affecté par une instruction INSERT dans l'autre. En d'autres termes, supposons que vous exécutez les instructions suivantes en série, dans n'importe quel ordre. Dans chaque cas, le résultat est l'une des instructions SELECT renvoyant une ligne de plus que si les transactions avaient été exécutées de manière simultanée. Il n'existe aucun ordre dans lequel les opérations peuvent s'exécuter en série et produire le même résultat que si elles étaient exécutées de manière simultanée. Ainsi, la dernière opération exécutée entraîne une erreur d'isolement sérialisable.

```
Session1_Redshift=# select * from tab1;
Session1_Redshift=# insert into tab2 values (1);
```

```
Session2_Redshift=# insert into tab1 values (1);
Session2_Redshift=# select * from tab2;
```

  Dans de nombreux cas, le résultat de l'instruction SELECT n'est pas important. En d'autres termes, l'atomicité des opérations dans les transactions n'est pas important. Dans ces cas-là, déplacez les instructions SELECT hors de leurs transactions, comme illustré dans les exemples suivants.

```
Session1_Redshift=# begin;
Session1_Redshift=# insert into tab1 values (1)
Session1_Redshift=# end;
```

```
Session1_Redshift=# select * from tab2;
```

```
Session2_Redshift # select * from tab1;
Session2_Redshift=# begin;
Session2_Redshift=# insert into tab2 values (1)
Session2_Redshift=# end;
```

Dans ces exemples, il n'y a aucune référence croisée dans les transactions. Les deux instructions INSERT n'ont aucun effet l'une sur l'autre. Dans ces exemples, il existe au moins un ordre dans lequel les transactions peuvent s'exécuter en série et produire le même résultat que si elle étaient exécutées de manière simultanée. Cela signifie que les transactions sont sérialisables.

- Forcez la sérialisation en verrouillant toutes les tables de chaque séance.

  La commande LOCK bloque les opérations pouvant entraîner des erreurs d'isolement sérialisable. Lorsque vous utilisez la commande LOCK, veillez à procéder comme suit :

  - Verrouillez toutes les tables affectées par la transaction, notamment celles affectées par les instructions SELECT en lecture seule à l'intérieur de la transaction.
  - Verrouillez les tables dans le même ordre, quel que soit l'ordre dans lequel ces opérations sont exécutées.
  - Verrouillez toutes les tables au début de la transaction, avant d'exécuter la moindre opération.
- Utiliser l'isolation des instantanés pour les transactions simultanées

  Utilisez une commande ALTER DATABASE avec l'isolation des instantanés. Pour plus d'informations sur le paramètre SNAPSHOT pour ALTER DATABASE, consultez Paramètres.

ERROR:1018 DETAIL : La relation n'existe pas

Lorsque vous exécutez des opérations Amazon Redshift simultanément dans différentes séances, un message d'erreur similaire au suivant s'affiche.

```
ERROR: 1018 DETAIL: Relation does not exist.
```

Les transactions dans Amazon Redshift suivent l'isolement des instantanés. Lorsqu'une transaction a démarré, Amazon Redshift prend un instantané de la base de données. Pour tout le cycle de vie de la transaction, la transaction est exécutée sur l'état de la base de données tel qu'il est reflété dans l'instantané. Si la transaction effectue la lecture à partir d'une table qui n'existe pas dans l'instantané, elle renvoie le message d'erreur 1018 affiché précédemment. Même lorsqu'une autre transaction

simultanée crée une table après que la transaction a pris l'instantané, la transaction ne peut pas effectuer la lecture à partir de la table créée.

Pour corriger cette erreur d'isolement de sérialisation, vous pouvez essayer de faire démarrer la transaction à un moment où vous savez que la table existe.

Si la table est créée par une autre transaction, ce moment commence au moins après que cette transaction a été validée. Assurez-vous également qu'aucune transaction simultanée qui aurait pu supprimer la table n'a été validée.

```
session1 = # BEGIN;
session1 = # DROP TABLE A;
session1 = # COMMIT;
```

```
session2 = # BEGIN;
```

```
session3 = # BEGIN;
session3 = # CREATE TABLE A (id INT);
session3 = # COMMIT;
```

```
session2 = # SELECT * FROM A;
```

La dernière opération exécutée en tant qu'opération de lecture par session2 entraîne une erreur d'isolement sérialisable. Cette erreur se produit lorsqu'une session2 prend un instantané et que la table a déjà été supprimée par une session1 validée. En d'autres termes, même si une session3 simultanée a créé la table, la session2 ne voit pas la table car elle ne se trouve pas dans l'instantané.

Pour résoudre cette erreur, vous pouvez réorganiser les séances comme suit.

```
session1 = # BEGIN;
session1 = # DROP TABLE A;
session1 = # COMMIT;
```

```
session3 = # BEGIN;
session3 = # CREATE TABLE A (id INT);
session3 = # COMMIT;
```

```
session2 = # BEGIN;
```

```
session2 = # SELECT * FROM A;
```

Lorsque la session2 prend son instantané, la session3 a déjà été validée et la table se trouve dans la base de données. La session2 peut effectuer la lecture à partir de la table sans aucune erreur.

Écriture et read/write opérations

Vous pouvez gérer le comportement spécifique des opérations d'écriture simultanées en décidant quand et comment exécuter différents types de commandes. Les commandes suivantes sont pertinentes pour cette discussion :

- Les commandes COPY, qui effectuent les chargements (initiaux ou incrémentiels)
- Les commandes INSERT qui ajoutent une ou plusieurs lignes à la fois
- Les commandes UPDATE, qui modifient les lignes existantes
- Les commandes DELETE, qui suppriment des lignes

Les opérations COPY et INSERT sont de pures opérations d'écriture. Les opérations DELETE et UPDATE sont read/write des opérations (pour que les lignes soient supprimées ou mises à jour, elles doivent d'abord être lues). Les résultats des opérations d'écriture simultanées dépendent des commandes spécifiques qui sont exécutées simultanément.

Les opérations UPDATE et DELETE se comportent différemment, car elles s'appuient sur une table initiale lue avant toute écriture. Étant donné que les transactions simultanées sont invisibles les unes pour UPDATEs les autres, les deux DELETEs doivent lire un instantané des données du dernier commit. Lorsque la première opération UPDATE ou DELETE libère son verrou, la deuxième opération UPDATE ou DELETE doit déterminer si les données qu'il va utiliser sont potentiellement obsolètes. Elles ne le sont pas, car la deuxième transaction n'obtient pas son instantané des données tant que la première transaction n'a pas libéré son verrou.

Situation d'impasse potentielle pour les transactions d'écriture simultanées impliquant plusieurs tables

Lorsque les transactions impliquent la mise à jour de plusieurs tables, il est toujours possible que des transactions exécutées simultanément soient bloquées lorsqu'elles essaient toutes deux d'écrire dans le même ensemble de tables. Une transaction libère tous ses verrous de table en même temps lorsqu'elle est validée ou annulée ; elle ne renonce pas aux verrous un par un.

Par exemple, supposons que les transactions T1 et T2 commencent approximativement en même temps. Si T1 commence à écrire dans la table A et T2 commence à écrire dans la table B, les deux transactions peuvent se poursuivre sans conflit. Cependant, si T1 finit d'écrire dans la table A et doit commencer à écrire dans la table B, il ne pourra pas continuer car T2 maintient toujours le verrou B. De même, si T2 finit d'écrire dans la table B et doit commencer à écrire dans la table A, il ne pourra pas non plus continuer car T1 maintient toujours le verrou A. Aucune transaction ne peut libérer ses verrous tant que toutes ses opérations d'écriture n'ont pas été validées, aucune des transactions ne peut continuer. Pour éviter ce type de blocage, vous devez planifier avec soin les opérations d'écriture simultanées. Par exemple, vous devez toujours mettre à jour les tables dans le même ordre des transactions et, si vous spécifiez des verrous, verrouillez les tables dans le même ordre avant d'effectuer les opérations DML.

Situation de blocage potentielle pour les transactions d'écriture simultanées impliquant une seule table

Dans un environnement d'isolation de snapshots, des blocages peuvent se produire lors de l'exécution de transactions d'écriture simultanées sur la même table. Le blocage de l'isolation des instantanés se produit lorsque des instructions INSERT ou COPY simultanées partagent un verrou et progressent, et qu'une autre instruction doit effectuer une opération (opération UPDATE, DELETE, MERGE ou DDL) nécessitant un verrou exclusif sur la même table.

Réfléchissez au scénario suivant :

Transaction 1 (T1) :

```
INSERT/COPY INTO table_A;
```

Transaction 2 (T2) :

```
INSERT/COPY INTO table_A;
            <UPDATE/DELETE/MERGE/DDL statement> table_A
```

Un blocage peut se produire lorsque plusieurs transactions comportant des opérations INSERT ou COPY sont exécutées simultanément sur la même table avec un verrou partagé, et que l'une de ces transactions suit son opération d'écriture pure par une opération nécessitant un verrou exclusif, telle qu'une instruction UPDATE, MERGE, DELETE ou DDL.

Pour éviter les blocages dans ces situations, vous pouvez séparer les instructions nécessitant un verrouillage exclusif (UPDATE/MERGE/DELETE/DDL statements) to a different transaction so that

any INSERT/COPY statements can progress simultaneously, and the statements requiring exclusive locks can execute after them. Alternatively, for transactions with INSERT/COPY statements and MERGE/UPDATE/MERGEinstructions) sur la même table. Vous pouvez inclure une logique de nouvelle tentative dans vos applications pour contourner les blocages potentiels.

Exemples d’écritures simultanées

Les exemples de pseudo-code suivants montrent comment les transactions se poursuivent ou attendent quand elles sont exécutées simultanément.

Exemples d'écriture simultanée avec isolation sérialisable

Opérations COPY simultanées dans la même table avec isolation sérialisable

La transaction 1 copie les lignes dans la table LISTING :

```
begin;
copy listing from ...;
end;
```

La transaction 2 commence simultanément dans une séance distincte et tente de copier de nouvelles lignes dans la table LISTING. La transaction 2 doit attendre que la transaction 1 libère le verrou en écriture sur la table LISTING, puis peut poursuivre.

```
begin;
[waits]
copy listing from ;
end;
```

Le même comportement se produirait si l’une d’une transactions ou les deux contenaient une commande INSERT au lieu d’une commande COPY.

Opérations DELETE simultanées à partir de la même table avec isolation sérialisable

La transaction 1 supprime des lignes d’une table :

```
begin;
delete from listing where ...;
end;
```

La transaction 2 commence simultanément et tente de supprimer des lignes de la même table. Elle réussit, car elle attend que la transaction 1 ait terminé avant de tenter de supprimer des lignes.

```
begin
[waits]
delete from listing where ;
end;
```

Le même comportement se produit si l'une des transactions ou les deux contiennent une commande UPDATE sur la même table au lieu d'une commande DELETE.

Transactions simultanées avec un mélange d'opérations de lecture et d'écriture avec isolation sérialisable

Dans cet exemple, la transaction 1 supprime des lignes de la table USERS, recharge la table, exécute une requête COUNT(*), puis une commande ANALYZE, avant de valider :

```
begin;
delete one row from USERS table;
copy ;
select count(*) from users;
analyze ;
end;
```

Pendant ce temps, la transaction 2 démarre. Cette transaction tente de copier les lignes supplémentaires dans la table USERS, analyse la table, puis exécute la même requête COUNT(*) comme première transaction :

```
begin;
[waits]
copy users from ...;
select count(*) from users;
analyze;
end;
```

La deuxième transaction réussit, car elle doit attendre que la première soit terminée. Sa requête COUNT renvoie le nombre en fonction de la charge qu'il a terminée.

Exemples d'écriture simultanée avec isolation par capture instantanée

Opérations COPY simultanées dans la même table avec isolation par capture instantanée

La transaction 1 copie les lignes dans la table LISTING :

```
begin;
copy listing from ...;
end;
```

La transaction 2 commence simultanément dans une séance distincte et tente de copier de nouvelles lignes dans la table LISTING. La transaction 2 peut progresser simultanément jusqu'à ce que l'une des transactions ait besoin d'écrire des données dans la table cible`listing`, date à laquelle elle s'exécutera de manière séquentielle.

```
begin;
//When the COPY statement from T1 needs to write data to the table, the COPY statement
 from T2 waits.
copy listing from ...;
end;
```

Le même comportement se produirait si l’une d’une transactions ou les deux contenaient une commande INSERT au lieu d’une commande COPY.

Opérations DELETE simultanées à partir de la même table avec isolation par capture instantanée

Les opérations DELETE ou UPDATE simultanées à partir de la même table avec isolation par capture instantanée s'exécutent de la même manière que les opérations exécutées avec isolation sérialisable.

Transactions simultanées avec un mélange d'opérations de lecture et d'écriture avec isolation instantanée

Les transactions simultanées exécutées avec des combinaisons d'opérations avec isolation instantanée s'exécutent de la même manière que les transactions avec des combinaisons d'opérations exécutées avec une isolation sérialisable.

Didacticiel : chargement des données à partir d'Amazon S3

Dans ce didacticiel, vous examinerez le processus de chargement de données dans vos tables de base de données Amazon Redshift depuis les fichiers de données d'un compartiment Amazon S3 de bout en bout.

Dans ce didacticiel, vous allez effectuer les opérations suivantes :

- Téléchargez des fichiers de données qui utilisent des formats CSV, délimités par un caractère et à largeur fixe.
- Créez un compartiment Amazon S3 pour contenir vos fichiers de données, puis chargez les fichiers de données dans le compartiment.
- Lancez un cluster Amazon Redshift et créez des tables de base de données ;
- Utilisez les commandes COPY pour charger les tables depuis les fichiers de données sur Amazon S3 ;
- Résolvez les erreurs de chargement et modifiez vos commandes COPY pour corriger les erreurs.

Prérequis

Vous avez besoin des prérequis suivants :

- Un AWS compte pour lancer un cluster Amazon Redshift et créer un compartiment dans Amazon S3.
- Vos AWS informations d'identification (rôle IAM) pour charger les données de test depuis Amazon S3. Si vous avez besoin d'un nouveau rôle IAM, consultez Création de rôles IAM.
- Client SQL tel que l'éditeur de requêtes de la console Amazon Redshift.

Ce didacticiel est conçu pour se suffire à lui-même. En plus de ce didacticiel, nous vous recommandons de suivre les didacticiels suivants pour avoir une compréhension plus complète de la conception et de l'utilisation des bases de données Amazon Redshift :

- Le Guide de démarrage d'Amazon Redshift vous explique le processus de création d'un cluster Amazon Redshift et de chargement d'exemples de données.

Présentation

Vous pouvez ajouter des données à vos tables Amazon Redshift en utilisant une commande INSERT ou une commande COPY. À l'échelle et à la vitesse d'un entrepôt des données Amazon Redshift, la commande COPY est beaucoup plus rapide et plus efficace que les commandes INSERT.

La commande COPY utilise l'architecture de traitement massivement parallèle (MPP) Amazon Redshift pour lire et charger des données en parallèle depuis plusieurs sources de données. Vous pouvez charger depuis des fichiers de données dans Amazon S3, Amazon EMR ou n'importe quel hôte distant accessible via une connexion SSH (Secure Shell). Ou vous pouvez charger directement depuis une table Amazon DynamoDB.

Dans ce didacticiel, vous utilisez la commande COPY pour charger les données à partir d'Amazon S3. Bon nombre des principes présentés ici s'appliquent également au chargement depuis d'autres sources de données.

Pour en savoir plus sur l'utilisation de la commande COPY, consultez les ressources suivantes :

- Bonnes pratiques de chargement des données sur Amazon Redshift
- Chargement de données à partir d'Amazon EMR
- Chargement des données à partir des hôtes distants
- Chargement de données à partir d'une table Amazon DynamoDB

Étape 1 : créer un cluster

Si vous disposez déjà d'un cluster que vous souhaitez utiliser, vous pouvez ignorer cette étape.

Pour les exercices de ce didacticiel, vous utilisez un cluster à quatre nœuds.

Pour créer un cluster

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/

   Dans le menu de navigation, choisissez le Tableau de bord des clusters provisionnés.

> **Important**
>
> Veillez à ce que vous disposiez des autorisations nécessaires pour exécuter les opérations de cluster. Pour plus d'informations sur l'octroi des autorisations nécessaires, consultez Autoriser Amazon Redshift à accéder aux AWS services.

2. En haut à droite, choisissez la AWS région dans laquelle vous souhaitez créer le cluster. Dans le cadre de ce didacticiel, sélectionnez USA Ouest (Oregon).
3. Dans le menu de navigation, choisissez Clusters, puis choisissez Créer un cluster. La page Créer un cluster s'affiche.
4. Dans la page Créer un cluster, saisissez les paramètres de votre cluster. Choisissez vos propres valeurs pour les paramètres, sauf pour modifier les valeurs suivantes :
   - Choisissez **`dc2.large`** pour le type de nœud.
   - Choisissez **`4`** pour le nombre de nœuds.
   - Dans la section Autorisations de cluster, choisissez un rôle IAM dans Rôles IAM disponibles. Ce rôle doit avoir été créé précédemment et avoir accès à Amazon S3. Choisissez ensuite Associate IAM role (Associer un rôle IAM) pour l'ajouter à la liste des Rôles IAM associés pour le cluster.
5. Choisissez Créer un cluster.

Suivez les étapes du Guide de démarrage d'Amazon Redshift pour vous connecter à votre cluster à partir d'un client SQL et tester une connexion. Vous n'avez pas besoin de suivre les dernières étapes de la mise en route pour créer des tables, télécharger des données et essayer des exemples de requêtes.

Étape 2 : Télécharger les fichiers de données

Au cours de cette étape, vous téléchargez un ensemble d'exemples de fichiers de données sur votre ordinateur. Dans l'étape suivante, vous chargez les fichiers dans un compartiment Amazon S3.

Pour télécharger les fichiers de données

1. Téléchargez le fichier compressé : LoadingDataSampleFiles.zip.
2. Extrayez les fichiers dans un dossier sur votre ordinateur.
3. Vérifiez que votre dossier contient les fichiers suivants.

```
customer-fw-manifest
customer-fw.tbl-000
customer-fw.tbl-000.bak
customer-fw.tbl-001
customer-fw.tbl-002
customer-fw.tbl-003
customer-fw.tbl-004
customer-fw.tbl-005
customer-fw.tbl-006
customer-fw.tbl-007
customer-fw.tbl.log
dwdate-tab.tbl-000
dwdate-tab.tbl-001
dwdate-tab.tbl-002
dwdate-tab.tbl-003
dwdate-tab.tbl-004
dwdate-tab.tbl-005
dwdate-tab.tbl-006
dwdate-tab.tbl-007
part-csv.tbl-000
part-csv.tbl-001
part-csv.tbl-002
part-csv.tbl-003
part-csv.tbl-004
part-csv.tbl-005
part-csv.tbl-006
part-csv.tbl-007
```

Étape 3 : Charger les fichiers dans un compartiment Amazon S3

Au cours de cette étape, vous allez créer un compartiment Amazon S3 et chargez les fichiers de données dans le compartiment.

Pour charger les fichiers dans un compartiment Amazon S3

1. Créez un compartiment dans Amazon S3.

   Pour plus d'informations sur la création d'un compartiment, consultez Création d'un compartiment dans le Guide de l'utilisateur Amazon Simple Storage Service.

a. Connectez-vous à la console Amazon S3 AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/s3/.

b. Choisissez Créer un compartiment.

c. Choisissez un Région AWS.

   Créez le compartiment dans la même région que votre cluster. Si votre cluster se trouve dans la région USA Ouest (Oregon), choisissez USA Ouest (Oregon) (us-west-2).

d. Dans la zone Nom du compartiment de la boîte de dialogue Créer un compartiment, entrez un nom de compartiment.

   Le nom de compartiment que vous choisissez doit être unique parmi tous les noms de compartiment existants dans Amazon S3. Afin de garantir cette unicité, vous pouvez ajouter le nom de votre organisation en préfixe du nom de vos compartiments. Les noms de compartiment doivent respecter certaines règles. Pour plus d'informations, consultez Limites et restrictions applicables aux compartiments dans le Guide de l'utilisateur Amazon Simple Storage Service.

e. Choisissez les valeurs par défaut recommandées pour les autres options.

f. Choisissez Créer un compartiment.

   Une fois le compartiment créé dans Amazon S3, celui-ci s'affiche dans la console, dans le volet Compartiments.

2. Créez un dossier.

   a. Choisissez le nom du nouveau compartiment.

   b. Choisissez le bouton Créer un dossier.

   c. Nommez le nouveau dossier **`load`**.

   > **Note**
   >
   > Le compartiment que vous avez créé ne figure pas dans un environnement de test (sandbox). Au cours de cet exercice, vous ajoutez des objets dans un véritable compartiment. Un montant nominal vous est facturé pour le temps de stockage des objets dans le compartiment. Pour plus d'informations sur les tarifs Amazon S3, consultez la page de Tarification Amazon S3.

3. Chargez les fichiers de données dans le nouveau compartiment Amazon S3.

a. Choisissez le nom du dossier de données.

b. Dans l'assistant Charger, choisissez Ajouter des fichiers.

   Suivez les instructions de la console Amazon S3 pour charger tous les fichiers que vous avez téléchargés et extraits.

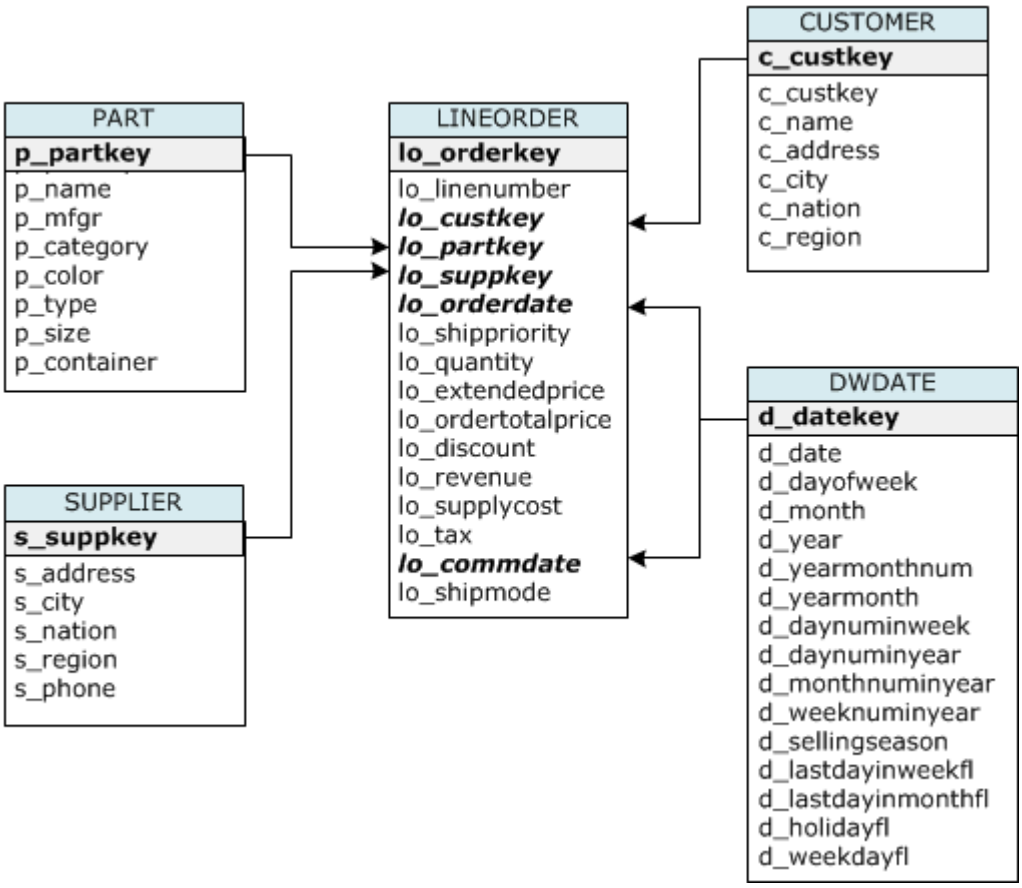
c. Choisissez Charger.

Informations d'identification de l'utilisateur

La commande COPY Amazon Redshift doit avoir un accès en lecture aux objets du fichier du compartiment Amazon S3. Si vous utilisez les mêmes informations d'identification de l'utilisateur pour créer le compartiment Amazon S3 que pour exécuter la commande COPY Amazon Redshift, celle-ci dispose de toutes les autorisations nécessaires. Si vous souhaitez utiliser les informations d'identification d'un autre utilisateur, vous pouvez accorder l'accès en utilisant les contrôles d'accès Amazon S3. La commande Amazon Redshift COPY nécessite au moins ListBucket des GetObject autorisations pour accéder aux objets du fichier dans le compartiment Amazon S3. Pour plus d'informations sur le contrôle de l'accès aux ressources Amazon S3, consultez Gestion des autorisations d'accès à vos ressources Amazon S3.

Étape 4 : Créer des exemples de tables

Pour ce didacticiel, vous utiliserez un ensemble de tables basé sur le schéma Star Schema Benchmark (SSB). Le diagramme suivant illustre le modèle de données SSB.



Il est possible que les tables SSB existent déjà dans la base de données actuelle. Le cas échéant, supprimez les tables de la base de données avant de les créer à l'aide des commandes CREATE TABLE de l'étape suivante. Les tables utilisées dans ce didacticiel peuvent avoir des attributs différents des tables existantes.

Pour créer des exemples de tables

1. Pour supprimer les tables SSB, exécutez les commandes suivantes dans votre client SQL.

```
drop table part cascade;
drop table supplier;
drop table customer;
drop table dwdate;
drop table lineorder;
```

2. Exécutez les commandes CREATE TABLE suivantes dans votre client SQL.

```
CREATE TABLE part
(
```

```
  p_partkey     INTEGER NOT NULL,
  p_name        VARCHAR(22) NOT NULL,
  p_mfgr        VARCHAR(6),
  p_category    VARCHAR(7) NOT NULL,
  p_brand1      VARCHAR(9) NOT NULL,
  p_color       VARCHAR(11) NOT NULL,
  p_type        VARCHAR(25) NOT NULL,
  p_size        INTEGER NOT NULL,
  p_container   VARCHAR(10) NOT NULL
);

CREATE TABLE supplier
(
  s_suppkey   INTEGER NOT NULL,
  s_name      VARCHAR(25) NOT NULL,
  s_address   VARCHAR(25) NOT NULL,
  s_city      VARCHAR(10) NOT NULL,
  s_nation    VARCHAR(15) NOT NULL,
  s_region    VARCHAR(12) NOT NULL,
  s_phone     VARCHAR(15) NOT NULL
);

CREATE TABLE customer
(
  c_custkey      INTEGER NOT NULL,
  c_name         VARCHAR(25) NOT NULL,
  c_address      VARCHAR(25) NOT NULL,
  c_city         VARCHAR(10) NOT NULL,
  c_nation       VARCHAR(15) NOT NULL,
  c_region       VARCHAR(12) NOT NULL,
  c_phone        VARCHAR(15) NOT NULL,
  c_mktsegment   VARCHAR(10) NOT NULL
);

CREATE TABLE dwdate
(
  d_datekey            INTEGER NOT NULL,
  d_date               VARCHAR(19) NOT NULL,
  d_dayofweek          VARCHAR(10) NOT NULL,
  d_month              VARCHAR(10) NOT NULL,
  d_year               INTEGER NOT NULL,
  d_yearmonthnum       INTEGER NOT NULL,
  d_yearmonth          VARCHAR(8) NOT NULL,
  d_daynuminweek       INTEGER NOT NULL,
```

```
  d_daynuminmonth      INTEGER NOT NULL,
  d_daynuminyear       INTEGER NOT NULL,
  d_monthnuminyear     INTEGER NOT NULL,
  d_weeknuminyear      INTEGER NOT NULL,
  d_sellingseason      VARCHAR(13) NOT NULL,
  d_lastdayinweekfl    VARCHAR(1) NOT NULL,
  d_lastdayinmonthfl   VARCHAR(1) NOT NULL,
  d_holidayfl          VARCHAR(1) NOT NULL,
  d_weekdayfl          VARCHAR(1) NOT NULL
);
CREATE TABLE lineorder
(
  lo_orderkey          INTEGER NOT NULL,
  lo_linenumber        INTEGER NOT NULL,
  lo_custkey           INTEGER NOT NULL,
  lo_partkey           INTEGER NOT NULL,
  lo_suppkey           INTEGER NOT NULL,
  lo_orderdate         INTEGER NOT NULL,
  lo_orderpriority     VARCHAR(15) NOT NULL,
  lo_shippriority      VARCHAR(1) NOT NULL,
  lo_quantity          INTEGER NOT NULL,
  lo_extendedprice     INTEGER NOT NULL,
  lo_ordertotalprice   INTEGER NOT NULL,
  lo_discount          INTEGER NOT NULL,
  lo_revenue           INTEGER NOT NULL,
  lo_supplycost        INTEGER NOT NULL,
  lo_tax               INTEGER NOT NULL,
  lo_commitdate        INTEGER NOT NULL,
  lo_shipmode          VARCHAR(10) NOT NULL
);
```

Étape 5 : Exécuter les commandes COPY

Vous exécutez des commandes COPY pour charger chacune des tables dans le schéma SSB. Les exemples de commande COPY illustrent le chargement dans différents formats, à l'aide de plusieurs options de la commande COPY, ainsi que la résolution des erreurs de chargement.

Syntaxe de la commande COPY

La syntaxe de base de la commande COPY est la suivante.

```
COPY table_name [ column_list ] FROM data_source CREDENTIALS access_credentials
 [options]
```

Pour exécuter une commande COPY, fournissez les valeurs suivantes.

Nom de la table

Table cible de la commande COPY. La table doit déjà exister dans la base de données. La table peut être temporaire ou permanente. La commande COPY ajoute les nouvelles données d'entrée à toutes les lignes existantes de la table.

Liste de colonnes

Par défaut, la commande COPY charge les champs depuis les données source dans les colonnes de la table dans l'ordre. Vous pouvez spécifier facultativement une liste de colonnes,, qui une liste séparée par des virgules des noms de colonnes, pour mapper les champs de données à des colonnes spécifiques. Vous n'utilisez pas de listes de colonnes de ce didacticiel. Pour plus d'informations, consultez Column List dans la référence de la commande COPY.

Source de données

Vous pouvez utiliser la commande COPY pour charger les données à partir d'un compartiment Amazon S3, un cluster Amazon EMR, un hôte distant à l'aide d'une connexion SSH, ou d'une table Amazon DynamoDB. Dans le cadre de ce didacticiel, vous chargez des fichiers de données dans un compartiment Amazon S3. Lorsque vous chargez depuis Amazon S3, vous devez fournir le nom du compartiment et l'emplacement des fichiers de données. Pour cela, fournissez soit un chemin d'accès à l'objet pour les fichiers de données ou l'emplacement du fichier manifeste qui répertorie explicitement chaque fichier de données et son emplacement.

- Préfixe de clé

  Un objet stocké dans Amazon S3 est identifié par la clé d'objet, qui inclut le nom du compartiment, les noms des dossiers, le cas échéant, et le nom de l'objet. Un préfixe de clé fait référence à un ensemble d'objets portant le même préfixe. Le chemin d'accès de l'objet est un préfixe de clé que la commande COPY utilise pour charger tous les objets partageant le même préfixe de clé. Par exemple, le préfixe de clé `custdata.txt` peut faire référence à un seul fichier ou à un ensemble de fichiers, notamment `custdata.txt.001`, `custdata.txt.002` et ainsi de suite.
- Fichier manifeste

Dans certains cas, vous aurez peut-être besoin de charger des fichiers avec des préfixes différentes, par exemple des fichiers provenant de plusieurs compartiments ou dossiers. Dans d'autres, vous devrez peut-être exclure des fichiers qui partagent un préfixe. Dans ces cas, vous pouvez utiliser un fichier manifeste. Un fichier manifeste répertorie explicitement chaque fichier de chargement et sa clé d'objet unique. Vous utilisez un fichier manifeste pour charger la table PART ultérieurement dans ce didacticiel.

Informations d'identification

Pour accéder aux AWS ressources contenant les données à charger, vous devez fournir les informations d'identification d' AWS accès à un utilisateur disposant de privilèges suffisants. Ces informations d'identification incluent un rôle IAM Amazon Resource Name (ARN). Pour charger des données depuis Amazon S3, les informations d'identification doivent inclure ListBucket des GetObject autorisations. Des informations d'identification supplémentaires sont nécessaires si vos données sont chiffrées. Pour plus d'informations, consultez Paramètres d'autorisation dans la référence de la commande COPY. Pour plus d'informations sur la gestion de l'accès, consultez Gestion des autorisations d'accès à vos ressources Amazon S3.

Options

Vous pouvez spécifier un certain nombre de paramètres avec la commande COPY afin de définir les formats de fichiers, gérer les formats de données, gérer les erreurs et contrôler d'autres fonctions. Dans ce didacticiel, vous utilisez les fonctions et les options de la commande COPY suivantes :

- Préfixe de clé

  Pour plus d'informations sur le chargement à partir de plusieurs fichiers en spécifiant un préfixe de clé, consultez Charger la table PART à l'aide de NULL AS.

- Format CSV

  Pour plus d'informations sur le chargement de données au format CSV, consultez Charger la table PART à l'aide de NULL AS.

- NULL AS

  Pour plus d'informations sur le chargement de PART à l'aide de l'option NULL AS, consultez Charger la table PART à l'aide de NULL AS.

- Format délimité par un caractère

Pour plus d'informations sur l'utilisation de l'option DELIMITER, consultez Les options DELIMITER et REGION.

- REGION

  Pour plus d'informations sur l'utilisation de l'option REGION, consultez Les options DELIMITER et REGION.

- Largeur de format fixe

  Pour plus d'informations sur le chargement de la table CUSTOMER à partir de données de largeur fixe, consultez Charger la table CUSTOMER à l'aide de MANIFEST.

- MAXERROR

  Pour plus d'informations sur l'utilisation de l'option MAXERROR, consultez Charger la table CUSTOMER à l'aide de MANIFEST.

- ACCEPTINVCHARS

  Pour plus d'informations sur l'utilisation de l'option ACCEPTINVCHARS, consultez Charger la table CUSTOMER à l'aide de MANIFEST.

- MANIFEST

  Pour plus d'informations sur l'utilisation de l'option MANIFEST, consultez Charger la table CUSTOMER à l'aide de MANIFEST.

- DATEFORMAT

  Pour plus d'informations sur l'utilisation de l'option DATEFORMAT, consultez Charger la table DWDATE à l'aide de DATEFORMAT.

- GZIP, LZOP et BZIP2

  Pour plus d'informations sur la compression de vos fichiers, consultez Charger plusieurs fichiers de données.

- COMPUPDATE

  Pour plus d'informations sur l'utilisation de l'option COMPUPDATE, consultez Charger plusieurs fichiers de données.

- Plusieurs fichiers

Pour plus d'informations sur le chargement de plusieurs fichiers, consultez Charger plusieurs fichiers de données.

Chargement des tables SSB

Vous utilisez les commandes COPY suivantes pour charger chacune des tables dans le schéma SSB. La commande de chaque table illustre les différentes options COPY et des techniques de résolution des problèmes.

Pour charger les tables SSB, procédez comme suit :

1. Remplacez le nom et les AWS informations d'identification du bucket
2. Charger la table PART à l'aide de NULL AS
3. Charger la table CUSTOMER à l'aide de MANIFEST
4. Charger la table DWDATE à l'aide de DATEFORMAT

Remplacez le nom et les AWS informations d'identification du bucket

Les commandes COPY de ce didacticiel sont présentées au format suivant.

```
copy table from 's3://<your-bucket-name>/load/key_prefix'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
options;
```

Pour chaque commande COPY, procédez de la façon suivante :

1. *<your-bucket-name>*Remplacez-le par le nom d'un compartiment situé dans la même région que votre cluster.

   Cette étape suppose que le compartiment et le cluster se situent dans la même région. Vous pouvez aussi spécifier la région à l'aide de l'option REGION de la commande COPY.
2. Remplacez *<aws-account-id>* et *<role-name>* par votre propre rôle Compte AWS et celui d'IAM. Le segment de la chaîne d'informations d'identification entre guillemets simples ne doit pas comporter d'espaces ou de sauts de ligne. Notez que le format de l'ARN peut être légèrement différent de celui de l'exemple. Il est préférable de copier l'ARN du rôle depuis la console IAM afin de garantir son exactitude lorsque vous exécutez les commandes COPY.

Charger la table PART à l'aide de NULL AS

Au cours de cette étape, vous utilisez les options CSV et NULL AS pour charger la table PART.

La commande COPY peut charger des données depuis plusieurs fichiers en parallèle, ce qui est beaucoup plus rapide que le chargement depuis un seul fichier. Pour illustrer ce principe, les données de chaque table de ce didacticiel sont divisées réparties dans huit fichiers, même si les fichiers sont très petits. Au cours d'une étape ultérieure, vous comparez la différence de temps entre le chargement d'un fichier unique et le chargement de plusieurs fichiers. Pour plus d'informations, consultez Chargement de fichiers de données.

Préfixe de clé

Vous pouvez charger à partir de plusieurs fichiers en spécifiant un préfixe de clé à l'ensemble de fichiers ou en répertoriant explicitement les fichiers dans un fichier manifeste. Au cours de cette étape, vous utilisez un préfixe de clé. Au cours d'une étape ultérieure, vous utilisez un fichier manifeste. Le préfixe de clé `'s3://amzn-s3-demo-bucket/load/part-csv.tbl'` charge l'ensemble de fichiers suivant dans le dossier `load`.

```
part-csv.tbl-000
part-csv.tbl-001
part-csv.tbl-002
part-csv.tbl-003
part-csv.tbl-004
part-csv.tbl-005
part-csv.tbl-006
part-csv.tbl-007
```

Format CSV

Le format CSV, qui signifie des valeurs séparées par une virgule, est un format couramment utilisé pour importer et exporter les données de feuille de calcul. Le format CSV est plus souple que le format séparé par une virgule, car il vous permet d'inclure des chaînes entre guillemets dans les champs. Le type de guillemets par défaut pour la commande COPY exécutée depuis le format CSV est celui des guillemets doubles ("), mais vous pouvez spécifier un autre type de guillemets à l'aide de l'option QUOTE AS. Lorsque vous utilisez les guillemets dans un champ, précédez le caractère de guillemets supplémentaires.

L'extrait suivant d'un fichier de données au format CSV pour la table PART illustre des chaînes entourées de guillemets (`"LARGE ANODIZED BRASS"`). Il montre également une chaîne entourée de doubles guillemets dans une chaîne entourée de guillemets (`"MEDIUM ""BURNISHED"" TIN"`).

```
15,dark sky,MFGR#3,MFGR#47,MFGR#3438,indigo,"LARGE ANODIZED BRASS",45,LG CASE
22,floral beige,MFGR#4,MFGR#44,MFGR#4421,medium,"PROMO, POLISHED BRASS",19,LG DRUM
23,bisque slate,MFGR#4,MFGR#41,MFGR#4137,firebrick,"MEDIUM ""BURNISHED"" TIN",42,JUMBO
 JAR
```

Les données de la table PART contiennent des caractères qui entraînent l'échec de la commande COPY. Dans cet exercice, vous allez résoudre les erreurs et les corriger.

Pour charger des données au format , ajoutez `csv`csv à votre commande COPY. Exécutez la commande suivante afin de charger la table PART.

```
copy part from 's3://<your-bucket-name>/load/part-csv.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
csv;
```

Vous pourriez recevoir un message d'erreur similaire au suivant.

```
An error occurred when executing the SQL command:
copy part from 's3://amzn-s3-demo-bucket/load/part-csv.tbl'
credentials' ...

ERROR: Load into table 'part' failed.  Check 'stl_load_errors' system table for
 details. [SQL State=XX000]

Execution time: 1.46s

1 statement(s) failed.
1 statement(s) failed.
```

Pour obtenir plus d'informations sur l'erreur, interrogez la table STL_LOAD_ERRORS. La requête suivante utilise la fonction SUBSTRING pour réduire les colonnes afin de faciliter la lecture et utilise LIMIT 10 pour réduire le nombre de lignes renvoyées. Vous pouvez ajuster les valeurs dans `substring(filename,22,25)` afin d'autoriser la longueur du nom de votre compartiment.

```
select query, substring(filename,22,25) as filename,line_number as line,
substring(colname,0,12) as column, type, position as pos, substring(raw_line,0,30) as
 line_text,
substring(raw_field_value,0,15) as field_text,
substring(err_reason,0,45) as reason
from stl_load_errors
order by query desc
```

```
limit 10;
```

```
 query  |    filename      | line |  column   |    type    | pos |
--------+------------------------+----------+-----------+------------+-----+----
 333765 | part-csv.tbl-000 |    1 |           |            |   0 |

 line_text        | field_text |                    reason
------------------+------------+-----------------------------------------------
 15,NUL next,     |            | Missing newline: Unexpected character 0x2c f
```

NULL AS

Les fichiers de données `part-csv.tbl` utilisent la marque de fin NUL (`\x000` ou `\x0`) pour indiquer des valeurs NULL.

> **Note**
>
> Malgré une orthographe très similaire, NUL et NULL ne sont pas identiques. NUL est un caractère UTF-8 avec le point de code `x000` qui est souvent utilisé pour indiquer la fin de l'enregistrement. NULL est une valeur SQL qui représente l'absence de données.

Par défaut, COPY traite une marque de fin NUL comme une marque de fin d'enregistrement et met fin à l'enregistrement, ce qui entraîne souvent des résultats inattendus ou une erreur. Il n'existe pas de méthode standard unique pour indiquer NULL dans les données de texte. Par conséquent, l'option de commande NULL AS COPY vous permet de spécifier le caractère à remplacer par NULL lors du chargement de la table. Dans cet exemple, vous voulez exécuter la commande COPY pour traiter la marque de fin NUL en tant que valeur NULL.

> **Note**
>
> La colonne de la table qui reçoit la valeur NULL doit être configurée comme acceptant la valeur null. Autrement dit, elle ne doit pas inclure la contrainte NOT NULL dans la spécification CREATE TABLE.

Pour charger la table PART en utilisant l'option AS NULL, exécutez la commande COPY suivante.

```
copy part from 's3://<your-bucket-name>/load/part-csv.tbl'
```

```
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
csv
null as '\000';
```

Pour vérifier que la commande COPY a chargé des valeurs NULL, exécutez la commande suivante pour sélectionner uniquement les lignes qui contiennent la valeur NULL.

```
select p_partkey, p_name, p_mfgr, p_category from part where p_mfgr is null;
```

```
 p_partkey |  p_name  | p_mfgr | p_category
-----------+----------+--------+------------
        15 | NUL next |        | MFGR#47
        81 | NUL next |        | MFGR#23
       133 | NUL next |        | MFGR#44
(2 rows)
```

Les options DELIMITER et REGION

Les options DELIMITER et REGION sont importantes pour comprendre comment charger les données.

Format délimité par un caractère

Les champs situés dans un fichier délimité par un caractère sont séparés par un caractère spécifique, comme une barre verticale (|), une virgule (,) ou une tabulation (\t). Les fichiers délimités par un caractère peuvent utiliser n'importe quel caractère ASCII unique, notamment l'un des caractères ASCII non affichables, comme le délimiteur. Vous spécifiez le délimiteur à l'aide de l'option DELIMITER. Le délimiteur par défaut est une barre verticale (|).

L'extrait suivant des données de la table SUPPLIER utilise le format délimité par une barre verticale.

```
1|1|257368|465569|41365|19950218|2-HIGH|0|17|2608718|9783671|4|2504369|92072|2|
19950331|TRUCK
1|2|257368|201928|8146|19950218|2-HIGH|0|36|6587676|9783671|9|5994785|109794|6|
19950416|MAIL
```

REGION

Dans la mesure du possible, vous devez localiser vos données de chargement dans la même AWS région que votre cluster Amazon Redshift. Si vos données et votre cluster se trouvent dans la même région, cela vous permet de réduire la latence et d'éviter des coûts de transfert régional de données.

Pour de plus amples informations, veuillez consulter Bonnes pratiques de chargement des données sur Amazon Redshift.

Si vous devez charger des données provenant d'une autre AWS région, utilisez l'option REGION pour spécifier la AWS région dans laquelle se trouvent les données de chargement. Si vous spécifiez une région, toutes les données de chargement, notamment les fichiers manifestes, doivent se situer dans la région désignée. Pour de plus amples informations, veuillez consulter REGION.

Par exemple, si votre cluster se trouve dans la région USA Est (Virginie du Nord) et que votre compartiment Amazon S3 se trouve dans la région USA Ouest (Oregon), la commande COPY suivante indique comment charger la table SUPPLIER à partir de données délimitées par des tuyaux.

```
copy supplier from 's3://amzn-s3-demo-bucket/ssb/supplier.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
delimiter '|'
gzip
region 'us-west-2';
```

Charger la table CUSTOMER à l’aide de MANIFEST

Au cours de cette étape, vous utilisez les options FIXEDWIDTH, MAXERROR, ACCEPTINVCHARS et MANIFEST pour charger la table CUSTOMER.

Les exemples de données de cet exercice contiennent des caractères qui entraînent des erreurs lorsque COPY tente de les charger. Vous utilisez l’option MAXERRORS et la table système STL_LOAD_ERRORS pour résoudre les erreurs de chargement, puis utiliser les options ACCEPTINVCHARS et MANIFEST pour éliminer les erreurs.

Format à largeur fixe

Le format à largeur fixe définit chaque champ en tant que nombre de caractères fixe, plutôt que comme des champs séparés par un délimiteur. L’extrait suivant des données de la table CUSTOMER utilise le format à largeur fixe.

```
1   Customer#000000001   IVhzIApeRb            MOROCCO  0MOROCCO  AFRICA      25-705
2   Customer#000000002   XSTf4,NCwDVaWNe6tE    JORDAN   6JORDAN   MIDDLE EAST 23-453
3   Customer#000000003   MG9kdTD               ARGENTINA5ARGENTINAAMERICA     11-783
```

L'ordre des label/width paires doit correspondre exactement à l'ordre des colonnes du tableau. Pour de plus amples informations, veuillez consulter FIXEDWIDTH.

La chaîne de spécification à largeur fixe pour les données de la table CUSTOMER est la suivante.

```
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
c_region :12, c_phone:15,c_mktsegment:10'
```

Pour charger la table CUSTOMER à partir de données à largeur fixe, exécutez la commande suivante.

```
copy customer
from 's3://<your-bucket-name>/load/customer-fw.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
 c_region :12, c_phone:15,c_mktsegment:10';
```

Vous devez recevoir un message d'erreur similaire au suivant.

```
An error occurred when executing the SQL command:
copy customer
from 's3://amzn-s3-demo-bucket/load/customer-fw.tbl'
credentials'...

ERROR: Load into table 'customer' failed.  Check 'stl_load_errors' system table for
 details. [SQL State=XX000]

Execution time: 2.95s

1 statement(s) failed.
```

MAXERROR

Par défaut, la première fois que COPY rencontre une erreur, la commande échoue et renvoie un message d'erreur. Pour gagner du temps pendant le test, vous pouvez utiliser l'option MAXERROR pour indiquer à COPY d'ignorer un nombre d'erreurs spécifié avant d'échouer. Du fait que nous anticipons des erreurs la première fois que nous testons le chargement des données de la table CUSTOMER, ajoutez `maxerror 10` à la commande COPY.

Pour effectuer le test en utilisant les options FIXEDWIDTH et MAXERROR, exécutez la commande suivante.

```
copy customer
from 's3://<your-bucket-name>/load/customer-fw.tbl'
```

```
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
 c_region :12, c_phone:15,c_mktsegment:10'
maxerror 10;
```

Cette fois-ci, au lieu d'un message d'erreur, vous recevez un message d'avertissement semblable au suivant.

```
Warnings:
Load into table 'customer' completed, 112497 record(s) loaded successfully.
Load into table 'customer' completed, 7 record(s) could not be loaded.  Check
 'stl_load_errors' system table for details.
```

L'avertissement indique que la commande COPY a rencontré sept erreurs. Pour vérifier les erreurs, interrogez la table STL_LOAD_ERRORS, comme illustré dans l'exemple suivant.

```
select query, substring(filename,22,25) as filename,line_number as line,
substring(colname,0,12) as column, type, position as pos, substring(raw_line,0,30) as
 line_text,
substring(raw_field_value,0,15) as field_text,
substring(err_reason,0,45) as error_reason
from stl_load_errors
order by query desc, filename
limit 7;
```

Les résultats de la requête STL_LOAD_ERRORS doivent être similaires à ce qui suit.

```
 query  |         filename          | line |  column   |    type    | pos |
 line_text           | field_text |              error_reason
--------+---------------------------+------+-----------
+------------+-----+-------------------------------+------------
+--------------------------------------------
 334489 | customer-fw.tbl.log       |    2 | c_custkey | int4       |  -1 | customer-
fw.tbl              | customer-f | Invalid digit, Value 'c', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    6 | c_custkey | int4       |  -1 | Complete
                  | Complete   | Invalid digit, Value 'C', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    3 | c_custkey | int4       |  -1 | #Total rows
                  | #Total row | Invalid digit, Value '#', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    5 | c_custkey | int4       |  -1 | #Status
                  | #Status    | Invalid digit, Value '#', Pos 0, Type: Integ
 334489 | customer-fw.tbl.log       |    1 | c_custkey | int4       |  -1 | #Load file
                  | #Load file | Invalid digit, Value '#', Pos 0, Type: Integ
```

```
 334489 | customer-fw.tbl000        |    1 | c_address | varchar    |  34 | 1
 Customer#000000001  | .Mayag.ezR | String contains invalid or unsupported UTF8
 334489 | customer-fw.tbl000        |    1 | c_address | varchar    |  34 | 1
 Customer#000000001  | .Mayag.ezR | String contains invalid or unsupported UTF8
(7 rows)
```

En examinant les résultats, vous pouvez voir qu'il y a deux messages dans la colonne `error_reasons` :

- ```
  Invalid digit, Value '#', Pos 0, Type: Integ
  ```

  Ces erreurs sont entraînées par le fichier `customer-fw.tbl.log`. Le problème est qu'il existe un fichier journal, pas un fichier de données et qu'il ne doit pas être chargé. Vous pouvez utiliser un fichier manifeste pour éviter le chargement du fichier incorrect.

- ```
  String contains invalid or unsupported UTF8
  ```

  Le type de données VARCHAR prend en charge les caractères UTF-8 de 3 octets au maximum. Si les données de chargement contiennent des caractères non pris en charge ou non valides, vous pouvez utiliser l'option ACCEPTINVCHARS pour remplacer chaque caractère non valide par un caractère alternatif spécifié.

Un autre problème se pose avec la charge qui est plus difficile à détecter : les résultats inattendus produits par la charge. Afin d'étudier ce problème, exécutez la commande suivante pour interroger la table CUSTOMER.

```
select c_custkey, c_name, c_address
from customer
order by c_custkey
limit 10;
```

```
 c_custkey |          c_name           |         c_address
-----------+---------------------------+---------------------------
         2 | Customer#000000002        | XSTf4,NCwDVaWNe6tE
         2 | Customer#000000002        | XSTf4,NCwDVaWNe6tE
         3 | Customer#000000003        | MG9kdTD
         3 | Customer#000000003        | MG9kdTD
         4 | Customer#000000004        | XxVSJsL
         4 | Customer#000000004        | XxVSJsL
```

```
         5 | Customer#000000005        | KvpyuHCplrB84WgAi
         5 | Customer#000000005        | KvpyuHCplrB84WgAi
         6 | Customer#000000006        | sKZz0CsnMD7mp4Xd0YrBvx
         6 | Customer#000000006        | sKZz0CsnMD7mp4Xd0YrBvx
(10 rows)
```

Les lignes doivent être uniques, mais il existe des doublons.

Pour vérifier les résultats inattendus, vous pouvez également vérifier le nombre de lignes qui ont été chargées. Dans notre cas, 100 000 lignes ont été chargées, mais le message de chargement a signalé le chargement de 112 497 enregistrements. Des lignes supplémentaires ont été chargées, car la commande COPY a chargé un fichier superflu, `customer-fw.tbl0000.bak`.

Dans cet exercice, vous utilisez un fichier manifeste pour éviter de charger des fichiers incorrects.

ACCEPTINVCHARS

Par défaut, lorsque la commande COPY rencontre un caractère qui n'est pas pris en charge par le type de données de la colonne, il ignore la ligne et renvoie une erreur. Pour plus d'informations sur les caractères UTF-8 non valides, consultez Erreurs de chargement de caractères multioctets.

Vous pouvez utiliser l'option MAXERRORS pour ignorer les erreurs et continuer le chargement, puis interroger STL_LOAD_ERRORS pour rechercher les caractères non valides et corriger les fichiers de données. Cependant, MAXERRORS est plus utile pour résoudre les problèmes de chargement et ne doit pas être utilisé, en général, dans un environnement de production.

L'option ACCEPTINVCHARS est généralement un choix plus adapté à la gestion des caractères non valides. ACCEPTINVCHARS indique que la commande COPY doit remplacer chaque caractère non valide par un caractère valide spécifié et poursuivre l'opération de chargement. Vous pouvez spécifier n'importe quel caractère ASCII valide, sauf NULL, comme caractère de remplacement. Le caractère de remplacement par défaut est un point d'interrogation (?). La commande COPY remplace les caractères de plusieurs octets par une chaîne de remplacement de la même longueur. Par exemple, un caractère de 4 octets serait remplacé par `'????'`.

COPY renvoie le nombre de lignes qui contenaient les caractères UTF-8 non valides. Cette commande ajoute également une entrée à la table système STL_REPLACEMENTS pour chaque ligne affectée, jusqu'à un maximum de 100 lignes par tranche de nœud. Les caractères UTF-8 non valides supplémentaires sont également remplacés, mais ces événements de remplacement ne sont pas enregistrés.

ACCEPTINVCHARS est valide uniquement pour les colonnes VARCHAR.

Pour cette étape, vous ajoutez ACCEPTINVCHARS au caractère de remplacement `'^'`.

MANIFEST

Lorsque vous exécutez la commande COPY à partir d'Amazon S3 à l'aide d'un préfixe de clé, il y a un risque que vous chargiez des tables indésirables. Par exemple, le dossier `'s3://amzn-s3-demo-bucket/load/` contient huit fichiers de données partageant le préfixe de clé `customer-fw.tbl` : `customer-fw.tbl0000`, `customer-fw.tbl0001` et ainsi de suite. Cependant, le même dossier contient également les fichiers superflus `customer-fw.tbl.log` et `customer-fw.tbl-0001.bak`.

Pour vous assurer que vous chargez tous les fichiers corrects et uniquement les fichiers corrects, utilisez un fichier manifeste. Le manifeste est un fichier texte au format JSON qui répertorie explicitement la clé d'objet unique de chaque fichier source à charger. Les objets de fichier peuvent se situer dans différents dossiers ou compartiments, mais ils doivent se trouver dans la même région. Pour plus d'informations, consultez MANIFEST.

L'exemple suivant illustre le texte `customer-fw-manifest`.

```
{
  "entries": [
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-000"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-001"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-002"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-003"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-004"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-005"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-006"},
    {"url":"s3://<your-bucket-name>/load/customer-fw.tbl-007"}
    ]
}
```

Pour charger les données de la table CUSTOMER en utilisant le fichier manifeste

1. Ouvrez le fichier `customer-fw-manifest` dans un éditeur de texte.
2. Remplacez *<your-bucket-name>* par le nom de votre compartiment.
3. Enregistrez le fichier.
4. Chargez le fichier dans le dossier de chargement de votre compartiment.
5. Exécutez la commande COPY suivante.

```
copy customer from 's3://<your-bucket-name>/load/customer-fw-manifest'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
fixedwidth 'c_custkey:10, c_name:25, c_address:25, c_city:10, c_nation:15,
 c_region :12, c_phone:15,c_mktsegment:10'
maxerror 10
acceptinvchars as '^'
manifest;
```

Charger la table DWDATE à l'aide de DATEFORMAT

Au cours de cette étape, vous utilisez les options DELIMITER et DATEFORMAT pour charger la table DWDATE.

Lors du chargement des colonnes DATE et TIMESTAMP, COPY attend le format par défaut, qui est YYYY-MM-DD pour les dates et YYYY-MM-DD HH:MI:SS pour les horodatages. Si les données de chargement n'utilisent pas un format par défaut, vous pouvez utiliser DATEFORMAT et TIMEFORMAT pour spécifier le format.

L'extrait suivant présente les formats de date du tableau DWDATE. Notez que les formats de date de la colonne deux sont incompatibles.

```
19920104 1992-01-04          Sunday  January 1992 199201 Jan1992 1 4 4 1...
19920112 January 12, 1992 Monday  January 1992 199201 Jan1992 2 12 12 1...
19920120 January 20, 1992 Tuesday     January 1992 199201 Jan1992 3 20 20 1...
```

DATEFORMAT

Vous ne pouvez spécifier qu'un seul format de date. Si les données de chargement contiennent des formats incompatibles, éventuellement dans différentes colonnes, ou si le format n'est pas connu au moment du chargement, vous utilisez DATEFORMAT avec l'argument `'auto'`. Lorsque `'auto'` est spécifié, la commande COPY reconnaît tous les formats d'heure ou de date valides et les convertira au format par défaut. L'option `'auto'` reconnaît plusieurs formats qui ne sont pas pris en charge lors de l'utilisation d'une chaîne DATEFORMAT et TIMEFORMAT. Pour plus d'informations, consultez Utilisation de la reconnaissance automatique avec DATEFORMAT et TIMEFORMAT.

Pour charger la table DWDATE, exécutez la commande COPY suivante.

```
copy dwdate from 's3://<your-bucket-name>/load/dwdate-tab.tbl'
```

```
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
delimiter '\t'
dateformat 'auto';
```

Charger plusieurs fichiers de données

Vous pouvez utiliser les options GZIP et COMPUPDATE pour charger une table.

Vous pouvez charger une table à partir d'un seul fichier de données ou de plusieurs fichiers. Procédez ainsi pour comparer les temps de chargement des deux méthodes.

GZIP, LZOP et BZIP2

Vous pouvez compresser vos fichiers à l’aide des formats de compression gzip, lzop ou bzip2. Lors du chargement de fichiers compressés, la commande COPY décompresse les fichiers pendant le processus de chargement. La compression de vos fichiers enregistre l’espace de stockage et réduit les temps de chargement.

COMPUPDATE

Lorsque la commande COPY charge une table vide sans aucun codage de compression, elle analyse les données de chargement pour déterminer les encodages optimaux. Elle modifie ensuite la table afin d’utiliser ces encodages avant de commencer le chargement. Ce processus d’analyse prend du temps, mais il se produit, tout au plus, une fois par table. Pour gagner du temps, vous pouvez ignorer cette étape en désactivant COMPUPDATE. Pour permettre une évaluation précise de temps de la commande COPY, vous désactivez COMPUPDATE pour cette étape.

Plusieurs fichiers

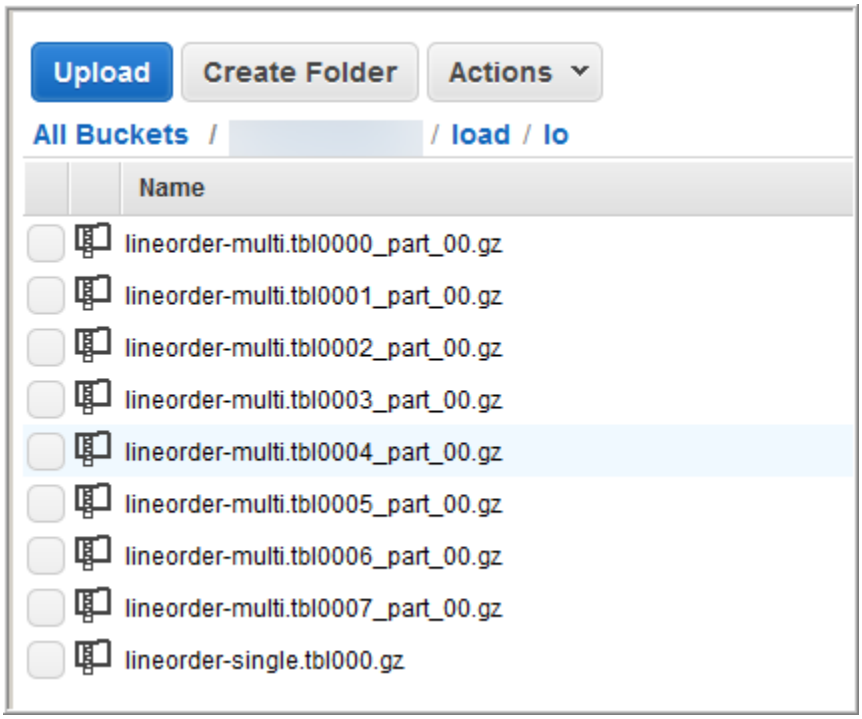
La commande COPY peut charger les données de manière très efficace en chargeant plusieurs fichiers en parallèle plutôt qu’un seul fichier. Vous pouvez scinder vos données en fichiers de telle sorte que le nombre de fichiers soit un multiple du nombre de tranches de votre cluster. Le cas échéant, Amazon Redshift divise la charge de travail et répartit les données de façon uniforme entre les tranches. Le nombre de tranches par nœud dépend de la taille de nœud du cluster. Pour plus d'informations sur le nombre de tranches pour chaque taille de nœud, consultez la rubrique À propos des clusters et nœuds dans le Guide de la gestion du cluster Amazon Redshift.

Par exemple, les nœuds de calcul de votre cluster dans ce didacticiel peuvent comporter deux tranches chacun, de sorte qu'un cluster à quatre nœuds comporte huit tranches. Au cours des étapes précédentes, les données de chargement étaient contenues dans huit fichiers, même si les fichiers

étaient très petits. Vous pouvez comparer la différence de temps entre le chargement à partir d'un seul fichier volumineux et le chargement à partir de plusieurs fichiers.

Même les fichiers contenant 15 millions d'enregistrements et occupant environ 1,2 Go sont très petits à l'échelle Amazon Redshift. Mais ils sont suffisants pour démontrer l'avantage en termes de performances du chargement à partir de plusieurs fichiers.

L'image suivante montre les fichiers de données pour LINEORDER.



Pour évaluer les performances de la commande COPY avec plusieurs fichiers

1. Lors d'un test en laboratoire, la commande suivante a été exécutée pour COPIER à partir d'un seul fichier. Cette commande affiche un bucket fictif.

```
copy lineorder from 's3://amzn-s3-demo-bucket/load/lo/lineorder-single.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
gzip
compupdate off
region 'us-east-1';
```

2. Les résultats ont été les suivants. Notez le délai d’exécution.

```
Warnings:
Load into table 'lineorder' completed, 14996734 record(s) loaded successfully.
```

```
0 row(s) affected.
copy executed successfully

Execution time: 51.56s
```

3. Ensuite, la commande suivante pour COPIER à partir de plusieurs fichiers a été exécutée.

```
copy lineorder from 's3://amzn-s3-demo-bucket/load/lo/lineorder-multi.tbl'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
gzip
compupdate off
region 'us-east-1';
```

4. Les résultats ont été les suivants. Notez le délai d'exécution.

```
Warnings:
Load into table 'lineorder' completed, 14996734 record(s) loaded successfully.

0 row(s) affected.
copy executed successfully

Execution time: 17.7s
```

5. Comparez les temps d'exécution.

   Dans notre expérience, le temps de chargement de 15 millions d'enregistrements est passé de 51,56 secondes à 17,7 secondes, soit une réduction de 65,7 %.

   Ces résultats sont basés sur l'utilisation d'un cluster à quatre nœuds. Si votre cluster comporte plusieurs nœuds, les gains de temps sont multipliés. Pour des clusters Amazon Redshift classiques, avec des dizaines, voire des centaines de nœuds, la différence est encore plus spectaculaire. Si vous disposez d'un cluster à nœud unique, il y a peu de différence entre les temps d'exécution.

Étape 6 : Vider et analyser la base de données

Chaque fois que vous ajoutez, supprimez ou modifiez un grand nombre de lignes, vous devez exécuter une commande VACUUM, puis une commande ANALYZE. Une commande vacuum récupère l'espace des lignes supprimées et restaure l'ordre de tri. La commande ANALYZE met à jour les métadonnées des statistiques, ce qui permet à l'optimiseur de requête de générer des plans

de requête plus précis. Pour plus d'informations, consultez Exécution de l'opération VACUUM sur les tables.

Si vous chargez les données dans l'ordre d'une clé de tri, le vidage est rapide. Dans ce didacticiel, vous avez ajouté un grand nombre de lignes, mais vous les avez ajoutées à des tables vides. Dans ce cas, il est inutile d'y recourir, et vous n'a pas besoin de supprimer toutes les lignes. COPY met automatiquement à jour les statistiques après le chargement d'une table vide. Vos statistiques devraient donc l'être up-to-date. Cependant, pour des raisons de bonne gestion, vous terminez ce didacticiel en effectuant un vidage et une analyse de votre base de données.

Pour vider et analyser la base de données, exécutez les commandes suivantes.

```
vacuum;
analyze;
```

Étape 7 : Nettoyer vos ressources

Votre cluster continue d'accumuler les frais aussi longtemps qu'il est en cours d'exécution. Une fois ce didacticiel terminé, vous devez rétablir votre environnement à l'état précédent en suivant les étapes décrites à l'Étape 5 : Annuler les droits d'accès et supprimer votre exemple de cluster du Guide de démarrage d'Amazon Redshift.

Si vous souhaitez conserver le cluster, mais que vous voulez récupérer le stockage utilisé par les tables SSB, exécutez les commandes suivantes.

```
drop table part;
drop table supplier;
drop table customer;
drop table dwdate;
drop table lineorder;
```

Suivant

Récapitulatif

Récapitulatif

Dans ce didacticiel, vous avez téléchargé les fichiers de données dans Amazon S3, puis utilisé les commandes COPY pour charger les données des fichiers dans des tables Amazon Redshift.

Vous avez chargé des données à l'aide des formats suivants :

- Délimité par un caractère
- CSV
- A largeur fixe

Vous avez utilisé la table système STL_LOAD_ERRORS pour résoudre les erreurs de chargement, puis utilisé les options REGION, MANIFEST, MAXERROR, ACCEPTINVCHARS, DATEFORMAT et NULL AS pour résoudre les erreurs.

Vous avez appliqué les bonnes pratiques suivantes au chargement des données :

- Utiliser une commande COPY pour charger les données
- Chargement de fichiers de données
- Utiliser une seule commande COPY pour charger à partir de plusieurs fichiers
- Compression de vos fichiers de données
- Vérifier les fichiers de données avant et après un chargement

Pour plus d'informations sur les bonnes pratiques Amazon Redshift, consultez les liens suivants :

- Bonnes pratiques de chargement des données sur Amazon Redshift
- Bonnes pratiques Amazon Redshift pour la conception de tables
- Bonnes pratiques Amazon Redshift pour la conception de requêtes

Déchargement de données dans Amazon Redshift

Pour décharger les données des tables de base de données sur un ensemble de fichiers d'un compartiment Amazon S3, vous pouvez utiliser la commande UNLOAD avec une instruction SELECT. Vous pouvez décharger les données texte en un format délimité ou un format largeur fixe, quel que soit le format des données utilisé pour le chargement. Vous pouvez également demander que soient créés des fichiers GZIP compressées.

Vous pouvez limiter l'accès des utilisateurs à votre compartiment Amazon S3 à l'aide d'informations d'identification de sécurité temporaires.

Rubriques

- Déchargement de données vers Amazon S3
- Déchargement de fichiers de données chiffrés
- Déchargement de données au format délimité ou au format à largeur fixe
- Rechargement de données déchargées

Déchargement de données vers Amazon S3

Amazon Redshift scinde les résultats d'une instruction SELECT sur un ensemble de fichiers, un ou plusieurs fichiers par tranche de nœud, afin de simplifier le rechargement parallèle des données. Sinon, vous pouvez spécifier que UNLOAD doit écrire les résultats en série sur un ou plusieurs fichiers en ajoutant l'option PARALLEL OFF. Vous pouvez limiter la taille des fichiers dans Amazon S3 en spécifiant le paramètre MAXFILESIZE. UNLOAD chiffre automatiquement les fichiers de données à l'aide du chiffrement côté serveur Amazon S3 (SSE-S3).

Vous pouvez utiliser dans la commande n'importe quelle instruction SELECT qu'Amazon Redshift prend en charge, à l'exception d'une instruction utilisant une clause LIMIT dans la sélection externe. Par exemple, vous pouvez utiliser une instruction SELECT qui inclut des colonnes spécifiques ou qui utilise une clause WHERE pour joindre plusieurs tables. Si votre requête contient des guillemets (encadrant les valeurs littérales, par exemple), vous devez les faire précéder d'une séquence d'échappement dans le texte de la requête (\'). Pour plus d'informations, consultez la référence de la commande SELECT. Pour plus d'informations sur l'utilisation d'une LIMIT, consultez Notes d'utilisation pour la commande UNLOAD.

Par exemple, la commande UNLOAD suivante envoie le contenu de la table VENUE au compartiment `s3://amzn-s3-demo-bucket/tickit/unload/` Amazon S3.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Les noms de fichiers créés par l'exemple précédent incluent le préfixe « `venue_` ».

```
venue_0000_part_00
venue_0001_part_00
venue_0002_part_00
venue_0003_part_00
```

Par défaut, UNLOAD écrit les données en parallèle dans plusieurs fichiers, selon le nombre de tranches du cluster. Pour écrire les données sur un seul fichier, spécifiez PARALLEL OFF. UNLOAD écrit les données en série, entièrement triées selon la clause ORDER BY, si elle est utilisée. La taille maximale d'un fichier de données est de 6,2 Go. Si la taille des données est supérieure à la valeur maximale, UNLOAD crée des fichiers supplémentaires, jusqu'à 6,2 Go chacun.

L'exemple suivant écrit le contenu de la table VENUE sur un seul fichier. Un seul fichier est obligatoire, car la taille du fichier est inférieure à 6,2 Go.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off;
```

> **Note**
>
> La commande UNLOAD est conçue pour utiliser le traitement parallèle. Nous vous recommandons de laisser PARALLEL activé dans la plupart des cas, surtout si les fichiers sont utilisés pour charger les tables à l'aide d'une commande COPY.

En supposant que la taille totale des données de VENUE est de 5 Go, l'exemple suivant écrit le contenu de VENUE dans 50 fichiers de 100 Mo chacun.

```
unload ('select * from venue')
```

```
to 's3://amzn-s3-demo-bucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off
maxfilesize 100 mb;
```

Si vous incluez un préfixe dans la chaîne du chemin Amazon S3, UNLOAD utilise ce préfixe pour les noms de fichier.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/unload/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Vous pouvez créer un fichier manifeste qui répertorie les fichiers de déchargement en spécifiant l'option MANIFEST dans la commande UNLOAD. Le manifeste est un fichier texte au format JSON qui répertorie explicitement l'URL de chaque fichier écrit sur Amazon S3.

L'exemple suivant inclut l'option MANIFEST.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest;
```

L'exemple suivant illustre un manifeste pour quatre fichiers de déchargement.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0000_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0001_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0002_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0003_part_00"}
  ]
}
```

Le fichier manifeste peut être utilisé pour charger les mêmes fichiers en utilisant une commande COPY avec l'option MANIFEST. Pour de plus amples informations, veuillez consulter Utilisation d'un manifeste pour spécifier les fichiers de données.

Une fois que vous avez terminé une opération UNLOAD, confirmez que les données ont été déchargées correctement en accédant au compartiment Amazon S3 sur lequel UNLOAD a écrit les

fichiers. Vous verrez un ou plusieurs fichiers numérotés par tranche, en commençant par le numéro zéro. Si vous avez spécifié l'option MANIFEST, vous verrez également un fichier se terminant par « `manifest` ». Par exemple :

```
amzn-s3-demo-bucket/tickit/venue_0000_part_00
amzn-s3-demo-bucket/tickit/venue_0001_part_00
amzn-s3-demo-bucket/tickit/venue_0002_part_00
amzn-s3-demo-bucket/tickit/venue_0003_part_00
amzn-s3-demo-bucket/tickit/venue_manifest
```

Vous pouvez obtenir par programmation une liste des fichiers qui ont été écrits dans Amazon S3 en appelant une opération de liste Amazon S3 une fois le DÉCHARGEMENT terminé. Vous pouvez également interroger STL_UNLO_LOG.

La requête suivante retourne le chemin d'accès des fichiers qui ont été créées par une opération UNLOAD. La fonction PG_LAST_QUERY_ID retourne la requête la plus récente.

```
select query, substring(path,0,40) as path
from stl_unload_log
where query=2320
order by path;

query |             path
-------+---------------------------------------
  2320 | s3://amzn-s3-demo-bucket/venue0000_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0001_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0002_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0003_part_00
(4 rows)
```

Si la quantité de données est très importante, Amazon Redshift peut scinder les fichiers en plusieurs parties par tranche. Par exemple :

```
venue_0000_part_00
venue_0000_part_01
venue_0000_part_02
venue_0001_part_00
venue_0001_part_01
venue_0001_part_02
...
```

Comme la commande UNLOAD suivante comprend une chaîne entre guillemets dans l'instruction SELECT, les guillemets sont précédés d'une séquence d'échappement (=\'0H\' ').

```
unload ('select venuename, venuecity from venue where venuestate=\'0H\' ')
to 's3://amzn-s3-demo-bucket/tickit/venue/ '
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Par défaut, UNLOAD échoue plutôt que de remplacer les fichiers existants dans le compartiment de destination. Pour remplacer les fichiers existants, y compris le fichier manifeste, spécifiez l'option ALLOWOVERWRITE.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest
allowoverwrite;
```

Déchargement de fichiers de données chiffrés

UNLOAD crée automatiquement des fichiers à l'aide du chiffrement côté serveur Amazon S3 avec des clés de chiffrement AWS gérées (SSE-S3). Vous pouvez également spécifier le chiffrement côté serveur avec une AWS Key Management Service clé (SSE-KMS) ou le chiffrement côté client avec une clé gérée par le client. UNLOAD ne prend pas en charge le chiffrement côté serveur Amazon S3 à l'aide d'une clé fournie par le client. Pour plus d'informations, consultez Protection des données à l'aide du chiffrement côté serveur.

Pour effectuer un déchargement vers Amazon S3 à l'aide d'un chiffrement côté serveur à l'aide d'une AWS KMS clé, utilisez le paramètre KMS_KEY_ID pour fournir l'ID de clé, comme indiqué dans l'exemple suivant.

```
unload ('select venuename, venuecity from venue')
to 's3://amzn-s3-demo-bucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
KMS_KEY_ID '1234abcd-12ab-34cd-56ef-1234567890ab'
encrypted;
```

Si vous souhaitez fournir votre propre clé de chiffrement, vous pouvez créer des fichiers de données chiffrés côté client dans Amazon S3 en utilisant la commande UNLOAD avec l'option ENCRYPTED. UNLOAD utilise le même processus de chiffrement par enveloppe que celui que le chiffrement

côté client Amazon S3 utilise. Vous pouvez ensuite utiliser la commande COPY avec l'option de chiffrement ENCRYPTED pour charger les fichiers chiffrés.

Voici comment cela fonctionne :

1. Vous créez une clé AES 256 bits codée en base64, que vous utiliserez comme clé de chiffrement privée, ou clé symétrique racine.
2. Vous entrez une commande UNLOAD qui inclut votre clé symétrique racine et l'option ENCRYPTED.
3. UNLOAD génère une clé one-time-use symétrique (appelée clé symétrique d'enveloppe) et un vecteur d'initialisation (IV), qu'il utilise pour chiffrer vos données.
4. UNLOAD chiffre la clé symétrique par enveloppe à l'aide de votre clé symétrique racine.
5. UNLOAD stocke ensuite les fichiers de données chiffrés dans Amazon S3 et stocke la clé d'enveloppe chiffrée et le vecteur d'initialisation comme métadonnées objet avec chaque fichier. La clé d'enveloppe chiffrée est stockée comme métadonnées objet `x-amz-meta-x-amz-key` et le vecteur d'initialisation est stocké comme métadonnées objet `x-amz-meta-x-amz-iv`.

Pour plus d'informations sur le processus de chiffrement des enveloppes, consultez l'article Chiffrement des données côté client avec le AWS SDK for Java et Amazon S3.

Pour décharger les fichiers de données chiffrés côté client, ajoutez la valeur de la clé racine à la chaîne des informations d'identification et incluez l'option ENCRYPTED. Si vous utilisez l'option MANIFEST, le fichier manifeste est également chiffré.

```
unload ('select venuename, venuecity from venue')
to 's3://amzn-s3-demo-bucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
manifest
encrypted;
```

Pour décharger des fichiers de données chiffrés compressés GZIP, incluez l'option GZIP, ainsi que la valeur de la clé racine et l'option ENCRYPTED.

```
unload ('select venuename, venuecity from venue')
to 's3://amzn-s3-demo-bucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
```

```
encrypted gzip;
```

Pour charger les fichiers de données chiffrés, ajoutez le paramètre MASTER_SYMMETRIC_KEY ayant la même valeur de clé racine et incluez l'option ENCRYPTED.

```
copy venue from 's3://amzn-s3-demo-bucket/encrypted/venue_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key '<root_key>'
encrypted;
```

Déchargement de données au format délimité ou au format à largeur fixe

Vous pouvez décharger les données au format délimité ou au format à largeur fixe. La sortie par défaut est délimitée par la barre verticale (|).

L'exemple suivant spécifie une virgule comme délimiteur :

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/venue/comma'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter ',';
```

Les fichiers de sortie obtenus se présentent ainsi :

```
20,Air Canada Centre,Toronto,ON,0
60,Rexall Place,Edmonton,AB,0
100,U.S. Cellular Field,Chicago,IL,40615
200,Al Hirschfeld Theatre,New York City,NY,0
240,San Jose Repertory Theatre,San Jose,CA,0
300,Kennedy Center Opera House,Washington,DC,0
...
```

Pour décharger le même jeu de résultats sur un fichier délimité par tabulation, entrez la commande suivante :

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/venue/tab'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
```

```
delimiter as '\t';
```

Sinon, vous pouvez utiliser une spécification FIXEDWIDTH. Cette spécification se compose d'un identificateur pour chaque colonne de table et de la largeur de la colonne (nombre de caractères). Comme la commande UNLOAD échoue plutôt que de tronquer les données, spécifiez une largeur au moins égale à l'entrée la plus longue de cette colonne. Le déchargement de données de largeur fixe fonctionne de manière similaire au déchargement de données délimitées, sauf que le résultat obtenu ne contient aucun caractère de délimitation. Par exemple :

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/tickit/venue/fw'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth '0:3,1:100,2:30,3:2,4:6';
```

Le résultat de la sortie de largeur fixe se présente comme suit :

```
20 Air Canada Centre         Toronto      ON0
60 Rexall Place              Edmonton     AB0
100U.S. Cellular Field       Chicago      IL40615
200Al Hirschfeld Theatre     New York CityNY0
240San Jose Repertory TheatreSan Jose     CA0
300Kennedy Center Opera HouseWashington   DC0
```

Pour plus d'informations sur les spécifications de FIXEDWIDTH, consultez la commande UNLOAD.

Rechargement de données déchargées

Pour recharger les résultats d'une opération de déchargement, vous pouvez utiliser une commande COPY.

L'exemple suivant illustre un cas simple dans lequel la table VENUE est déchargée à l'aide d'un fichier manifeste, tronquée et rechargée.

```
unload  ('select * from venue order by venueid')
to 's3://amzn-s3-demo-bucket/tickit/venue/reload_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest
delimiter '|';
```

```
truncate venue;

copy venue
from 's3://amzn-s3-demo-bucket/tickit/venue/reload_manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest
delimiter '|';
```

Une fois qu'elle a été à nouveau chargée, la table VENUE se présente comme suit :

```
select * from venue order by venueid limit 5;

 venueid |         venuename          |  venuecity  | venuestate | venueseats
---------+----------------------------+-------------+------------+-----------
       1 | Toyota Park                | Bridgeview  | IL         |          0
       2 | Columbus Crew Stadium      | Columbus    | OH         |          0
       3 | RFK Stadium                | Washington  | DC         |          0
       4 | CommunityAmerica Ballpark | Kansas City | KS         |          0
       5 | Gillette Stadium           | Foxborough  | MA         |      68756
(5 rows)
```

Fonctions définies par l'utilisateur dans Amazon Redshift

Vous pouvez créer une fonction scalaire personnalisée définie par l'utilisateur (UDF) à l'aide d'une clause SELECT SQL ou d'un programme Python. La nouvelle fonction est stockée dans la base de données ; elle est disponible pour tout utilisateur disposant de privilèges d'exécution suffisants. Vous exécutez une fonction UDF scalaire personnalisée de la même manière que vous exécutez les fonctions Amazon Redshift existantes.

Pour Python UDFs, en plus d'utiliser les fonctionnalités standard de Python, vous pouvez importer vos propres modules Python personnalisés. Pour de plus amples informations, veuillez consulter Support du langage Python pour UDFs. Notez que Python 3 n'est pas disponible pour Python UDFs. Pour obtenir le support de Python 3 pour Amazon Redshift UDFs, utilisez Lambda scalaire UDFs plutôt.

Vous pouvez également créer AWS Lambda UDFs des fonctions personnalisées définies dans Lambda dans le cadre de vos requêtes SQL. Lambda vous UDFs permet d'écrire des éléments complexes UDFs et de les intégrer à des composants tiers. Ils peuvent également vous aider à surmonter certaines des limites de Python et SQL actuels UDFs. Par exemple, elles peuvent vous aider à accéder aux ressources de réseau et de stockage et à écrire des instructions SQL plus complètes. Vous pouvez créer Lambda UDFs dans tous les langages de programmation pris en charge par Lambda, tels que Java, Go, Node.js, C# PowerShell, Python et Ruby. Vous pouvez également utiliser une exécution personnalisée.

Par défaut, tous les utilisateurs peuvent exécuter UDFs. Pour plus d’informations sur les privilèges, consultez Sécurité et autorisations UDF.

Rubriques

- Sécurité et autorisations UDF
- Prévention des conflits de dénomination des fonctions UDF
- SQL scalaire UDFs
- Python scalaire UDFs
- Lambda scalaire UDFs
- Exemples de cas d'utilisation pour les fonctions définies par l'utilisateur () UDFs

Sécurité et autorisations UDF

Pour créer une fonction définie par l'utilisateur, vous devez avoir l'autorisation pour USAGE ON LANGUAGE pour SQL ou plpythonu (Python). Par défaut, USAGE ON LANGUAGE SQL est accordé à PUBLIC, mais vous devez accorder explicitement USAGE ON LANGUAGE PLPYTHONU à des utilisateurs ou des groupes spécifiques.

Pour révoquer le privilège USAGE pour SQL, révoquez d'abord USAGE de PUBLIC. Accordez ensuite l'utilisation de SQL uniquement aux utilisateurs ou groupes spécifiques autorisés à créer du SQL UDFs. L'exemple suivant révoque le privilège USAGE pour SQL de PUBLIC. Ensuite, il accorde le privilège USAGE au groupe d'utilisateurs `udf_devs`.

```
revoke usage on language sql from PUBLIC;
grant usage on language sql to group udf_devs;
```

Pour exécuter une fonction UDF, vous devez disposer de l'autorisation d'exécution pour chaque fonction. Par défaut, l'autorisation d'exécuter new UDFs est accordée à PUBLIC. Pour limiter l'utilisation, révoquez cette autorisation de PUBLIC pour la fonction. Ensuite, accordez le privilège à des individus ou à des groupes spécifiques.

L'exemple suivant révoque le privilège d'exécution de la fonction `f_py_greater` de PUBLIC. Ensuite, il accorde le privilège USAGE au groupe d'utilisateurs `udf_devs`.

```
revoke execute on function f_py_greater(a float, b float) from PUBLIC;
grant execute on function f_py_greater(a float, b float) to group udf_devs;
```

Les super-utilisateurs ont tous les privilèges par défaut.

Pour plus d'informations, consultez GRANT et REVOKE.

Prévention des conflits de dénomination des fonctions UDF

Vous pouvez éviter les conflits potentiels et les résultats inattendus en prenant garde à vos conventions de dénomination de fonctions UDF avant la mise en œuvre. Etant donné que les noms de fonctions peuvent être surchargés, ils peuvent entrer en conflit avec des noms de fonctions Amazon Redshift existants et à venir. Cette rubrique traite de la surcharge et propose une stratégie permettant d'éviter les conflits.

Surcharge des noms de fonctions

Une fonction est identifiée par son nom et sa signature, c'est-à-dire le nombre d'arguments d'entrée et les types de données des arguments. Deux fonctions d'un même schéma peuvent porter le même nom si elles ont des signatures différentes. Autrement dit, les noms des fonctions peuvent être surchargés.

Lorsque vous exécutez une requête, le moteur de requête détermine quelle fonction appeler en fonction du nombre d'arguments que vous fournissez et des types de données des arguments. Vous pouvez utiliser une surcharge pour simuler des fonctions avec un nombre d'arguments variable, jusqu'à la limite autorisée par la commande CREATE FUNCTION.

Éviter les conflits avec les fonctions intégrées d'Amazon Redshift

Nous vous recommandons de les nommer tous UDFs à l'aide du préfixe`f_`. Amazon Redshift réserve le `f_` préfixe exclusivement pour UDFs et en préfixant vos noms UDF avec`f_`, vous vous assurez que votre nom UDF n'entrera en conflit avec aucun nom de fonction SQL intégré Amazon Redshift existant ou futur. Par exemple, en nommant une nouvelle fonction UDF `f_sum`, vous évitez tout conflit avec la fonction SUM Amazon Redshift. De même, si vous nommez une nouvelle fonction `f_fibonacci`, vous évitez les conflits si Amazon Redshift ajoute une fonction nommée FIBONACCI dans une version ultérieure.

Vous pouvez créer une fonction UDF avec les mêmes nom et signature qu'une fonction SQL intégrée Amazon Redshift existant sans que le nom de la fonction soit surchargé si la fonction UDF et la fonction intégrée existent dans différents schémas. Etant donné qu'il existe des fonctions intégrées dans le schéma catalogue système, pg_catalog, vous pouvez créer une fonction UDF portant le même nom dans un autre schéma, comme un schéma public ou un schéma défini par l'utilisateur. Dans certains cas, vous pouvez appeler une fonction qui n'est pas explicitement qualifiée par un nom de schéma. Si c'est le cas, Amazon Redshift recherche d'abord le schéma pg_catalog par défaut. Ainsi, une fonction intégrée s'exécute avant une nouvelle fonction UDF portant le même nom.

Vous pouvez modifier ce comportement en définissant le chemin de recherche de manière à placer pg_catalog à la fin. Dans ce cas, vous aurez la UDFs priorité sur les fonctions intégrées, mais cette pratique peut entraîner des résultats inattendus. L'adoption d'une stratégie d'attribution de noms unique, par exemple en utilisant le préfixe réservé `f_`, est une pratique plus fiable. Pour plus d'informations, consultez SET et search_path.

SQL scalaire UDFs

Une fonction scalaire SQL définie par l'utilisateur intègre une clause SELECT SQL qui s'exécute lorsque la fonction est appelée et renvoie une valeur unique. La commande CREATE FUNCTION définit les paramètres suivants :

- (Facultatif) Arguments d'entrée. Chaque argument doit disposer d'un type de données.
- Un type de données de retour.
- Une clause SELECT SQL. Dans la clause SELECT, faites référence aux arguments d'entrée à l'aide de $1, $2, et ainsi de suite, en fonction de l'ordre des arguments dans la définition de fonction.

Les types de données d'entrée et de retour peuvent être de n'importe quel type de données Amazon Redshift standard.

N'incluez pas de clause FROM dans la clause SELECT. A la place, incluez la clause FROM dans l'instruction SQL qui appelle la fonction SQL définie par l'utilisateur.

La clause SELECT ne peut pas inclure les types de clause suivants :

- FROM
- INTO
- WHERE
- GROUP BY
- ORDER BY
- LIMIT

Exemple de fonction scalaire SQL

L'exemple suivant crée une fonction qui compare deux nombres et renvoie la valeur la plus grande. Pour plus d'informations, consultez CREATE FUNCTION.

```
create function f_sql_greater (float, float)
  returns float
stable
as $$
```

```
  select case when $1 > $2 then $1
    else $2
  end
$$ language sql;
```

La requête suivante appelle la nouvelle fonction f_sql_greater pour interroger la table SALES et renvoyer COMMISSION ou 20 % du PRICEPAID, quelle que soit la valeur la plus grande.

```
select f_sql_greater(commission, pricepaid*0.20) from sales;
```

Python scalaire UDFs

Une fonction scalaire Python définie par l'utilisateur intègre un programme Python qui s'exécute lorsque la fonction est appelée et renvoie une valeur unique. La commande CREATE FUNCTION définit les paramètres suivants :

- (Facultatif) Arguments d'entrée. Chaque argument doit disposer d'un nom et d'un type de données.
- Un type de données de retour.
- Un programme Python exécutable.

Les types de données d'entrée et de retour pour Python UDFs peuvent être l'un des types suivants :

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISION
- BOOLEAN
- CHAR
- VARCHAR
- DATE
- TIMESTAMP
- N'IMPORTE QUEL ÉLÉMENT

Les alias de ces types sont également valides. Pour une liste complète des types de données et de leurs alias, consultezTypes de données.

Lorsque Python UDFs utilise le type de données ANYELEMENT, Amazon Redshift le convertit automatiquement en un type de données standard en fonction des arguments fournis lors de l'exécution. Pour de plus amples informations, veuillez consulter Type de données ANYELEMENT.

Lorsqu'une requête Amazon Redshift appelle une fonction UDF scalaire, les étapes suivantes se produisent au moment de l'exécution :

1. La fonction convertit les arguments d'entrée en types de données Python.

   Pour un mappage des types de données Amazon Redshift avec les types de données Python, consultez Types de données de fonctions Python définies par l'utilisateur.
2. La fonction exécute le programme Python, en transmettant les arguments d'entrée convertis.
3. Le code Python renvoie une valeur unique. Le type de données de la valeur de retour doit correspondre au type de données RETURNS spécifié par la définition de fonction.
4. La fonction convertit la valeur de retour Python dans le type de données Amazon Redshift, puis renvoie cette valeur à la requête.

> **Note**
>
> Python 3 n'est pas disponible pour Python UDFs. Pour obtenir le support de Python 3 pour Amazon Redshift UDFs, utilisez Lambda scalaire UDFs plutôt.

Exemple de fonction scalaire Python définie par l’utilisateur

L’exemple suivant crée une fonction qui compare deux nombres et renvoie la valeur la plus grande. Notez que la mise en retrait du code entre les signes de dollar doubles ($$) est une exigence Python. Pour plus d'informations, consultez CREATE FUNCTION.

```
create function f_py_greater (a float, b float)
  returns float
stable
as $$
  if a > b:
    return a
```

```
  return b
$$ language plpythonu;
```

La requête suivante appelle la nouvelle fonction `f_greater` pour interroger la table SALES et renvoyer COMMISSION ou 20 % du PRICEPAID, quelle que soit la valeur la plus grande.

```
select f_py_greater (commission, pricepaid*0.20) from sales;
```

Types de données de fonctions Python définies par l'utilisateur

Python UDFs peut utiliser n'importe quel type de données Amazon Redshift standard pour les arguments d'entrée et la valeur de retour de la fonction. Outre les types de données standard, prenez en UDFs charge le type de données ANYELEMENT, qu'Amazon Redshift convertit automatiquement en type de données standard en fonction des arguments fournis lors de l'exécution. Scalar UDFs peut renvoyer un type de données ANYELEMENT. Pour de plus amples informations, veuillez consulter Type de données ANYELEMENT.

Au cours de l'exécution, Amazon Redshift convertit les arguments des types de données Amazon Redshift en types de données Python pour pouvoir les traiter. Il convertit ensuite la valeur de retour du type de données Python en type de données Amazon Redshift correspondant. Pour plus d'informations sur les types de données Amazon Redshift, consultez Types de données.

Le tableau suivant mappe les types de données Amazon Redshift aux types de données Python.

Type de données Amazon Redshift	Type de données Python
smallint	int
integer	
bigint	
short	
long	
decimal ou numeric	decimal
double	float

Type de données Amazon Redshift	Type de données Python
real	
boolean	bool
char varchar	chaîne
timestamp	datetime

Type de données ANYELEMENT

ANYELEMENT est un type de données polymorphe. Cela signifie que si une fonction est déclarée à l'aide de ANYELEMENT pour un type de données d'argument, la fonction peut accepter tous les types de données Amazon Redshift standard comme entrée pour cet argument lorsque la fonction est appelée. L'argument ANYELEMENT est défini sur le type de données qui lui est réellement transmis lorsque la fonction est appelée.

Si une fonction utilise plusieurs types de données ANYELEMENT, ils doivent tous être résolus au même type de données réel lorsque la fonction est appelée. Tous les types de données d'argument ANYELEMENT sont définies sur le type de données réel du premier argument transmis à un ANYELEMENT. Par exemple, une fonction déclarée comme `f_equal(anyelement, anyelement)` prendra les deux valeurs d'entrée, tant qu'elles sont du même type de données.

Si la valeur de retour d'une fonction est déclarée comme ANYELEMENT, au moins un argument d'entrée doit être ANYELEMENT. Le type de données réel de la valeur de retour est identique au type de données réel fourni pour l'argument d'entrée ANYELEMENT.

Support du langage Python pour UDFs

Vous pouvez créer une fonction UDF personnalisée basée sur le langage de programmation Python. La bibliothèque standard Python 2.7 peut être utilisée dans UDFs, à l'exception des modules suivants :

- ScrolledText
- Tix
- Tkinter

- tk
- turtle
- smtpd

En plus de la bibliothèque standard Python, les modules suivants font partie de la mise en œuvre d'Amazon Redshift :

- numpy 1.8.2
- pandas 0.14.1
- python-dateutil 2.2
- pytz 2014.7
- scipy 0.12.1
- six 1.3.0
- wsgiref 0.1.2

Vous pouvez également importer vos propres modules Python personnalisés et les rendre utilisables en UDFs exécutant une CREATE LIBRARY commande. Pour de plus amples informations, veuillez consulter Exemple : importation de modules de bibliothèque Python personnalisés.

> **Important**
>
> Amazon Redshift bloque tout accès réseau et tout accès en écriture au système de fichiers via celui-ci. UDFs

> **Note**
>
> Python 3 n'est pas disponible pour Python UDFs. Pour obtenir le support de Python 3 pour Amazon Redshift UDFs, utilisez Lambda scalaire UDFs plutôt.

Exemple : importation de modules de bibliothèque Python personnalisés

Vous définissez les fonctions scalaires à l'aide de la syntaxe du langage Python. Vous pouvez utiliser les modules de la bibliothèque standard Python et les modules préinstallés d'Amazon Redshift. Vous

pouvez également créer vos propres modules de bibliothèque Python personnalisés et importer les bibliothèques dans vos clusters, ou utiliser des bibliothèques existantes de Python ou de tiers.

Vous ne pouvez pas créer une bibliothèque qui contient un module avec le même nom qu'un module de la bibliothèque standard Python ou qu'un module Amazon Redshift préinstallé. Si une bibliothèque existante installée par l'utilisateur emploie le même package Python qu'une bibliothèque que vous créez, vous devez supprimer la bibliothèque existante avant d'installer la nouvelle bibliothèque.

Vous devez être un super-utilisateur ou disposer du privilège `USAGE ON LANGUAGE plpythonu` pour installer des bibliothèques personnalisées. Toutefois, tous les utilisateurs disposant de privilèges suffisants pour créer des fonctions peuvent utiliser les bibliothèques installées. Vous pouvez interroger le catalogue système PG_LIBRARY pour afficher des informations sur les bibliothèques installées sur votre cluster.

Importation d'un module Python personnalisé dans votre cluster

Cette section fournit un exemple d'importation d'un module Python personnalisé dans votre cluster. Pour effectuer les étapes de cette section, vous devez disposer d'un compartiment Amazon S3, dans lequel vous téléchargez le package de la bibliothèque. Ensuite, vous installez le package dans votre cluster. Pour plus d'informations sur la création de compartiments, consultez Création d'un compartiment dans le Guide de l'utilisateur Amazon Simple Storage Service.

Dans cet exemple, supposons que vous créez UDFs pour travailler avec des positions et des distances dans vos données. Connectez-vous à votre cluster Amazon Redshift depuis un outil client SQL et exécutez les commandes suivantes pour créer les fonctions.

```
CREATE FUNCTION f_distance (x1 float, y1 float, x2 float, y2 float) RETURNS float
 IMMUTABLE as $$
    def distance(x1, y1, x2, y2):
        import math
        return math.sqrt((y2 - y1) ** 2 + (x2 - x1) ** 2)

    return distance(x1, y1, x2, y2)
$$ LANGUAGE plpythonu;

CREATE FUNCTION f_within_range (x1 float, y1 float, x2 float, y2 float) RETURNS bool
 IMMUTABLE as $$
    def distance(x1, y1, x2, y2):
        import math
        return math.sqrt((y2 - y1) ** 2 + (x2 - x1) ** 2)

    return distance(x1, y1, x2, y2) < 20
```

```
$$ LANGUAGE plpythonu;
```

Notez que quelques lignes de code sont identiques dans les fonctions précédentes. Cette duplication est nécessaire, car une fonction UDF ne peut pas faire référence au contenu d'une autre fonction UDF et que les deux fonctions nécessitent la même fonctionnalité. Cependant, au lieu de dupliquer le code dans plusieurs fonctions, vous pouvez créer une bibliothèque personnalisée et configurer vos fonctions pour l'utiliser.

Pour ce faire, commencez par créer le package de la bibliothèque en procédant comme suit :

1. Créez un dossier nommé geometry. Ce dossier constitue le package de niveau supérieur de la bibliothèque.
2. Dans le geometry, créez un fichier nommé `__init__.py`. Notez que le nom du fichier contient deux caractères de soulignement doubles. Ce fichier indique à Python que le package peut être initialisé.
3. Toujours dans le dossier geometry, créez un dossier nommé trig. Ce dossier constitue le sous-package de la bibliothèque.
4. Dans le dossier trig, créez un autre fichier nommé `__init__.py` et un fichier nommé `line.py`. Dans ce dossier, `__init__.py` indique à Python que le sous-package peut être initialisé et que `line.py` est le fichier qui contient le code de la bibliothèque.

   Votre structure de dossier et de fichier devrait être la même que la suivante :

   ```
   geometry/
      __init__.py
      trig/
         __init__.py
         line.py
   ```

   Pour plus d'informations sur la structure du package, consultez la section Modules du didacticiel Python sur le site web de Python.
5. Le code suivant contient une classe et des fonctions de membre pour la bibliothèque. Copiez et collez-le dans `line.py`.

   ```
   class LineSegment:
     def __init__(self, x1, y1, x2, y2):
       self.x1 = x1
       self.y1 = y1
   ```

```
    self.x2 = x2
    self.y2 = y2
  def angle(self):
    import math
    return math.atan2(self.y2 - self.y1, self.x2 - self.x1)
  def distance(self):
    import math
    return math.sqrt((self.y2 - self.y1) ** 2 + (self.x2 - self.x1) ** 2)
```

Une fois que vous avez créé le package, procédez comme suit pour préparer le package et chargez-le dans Amazon S3.

1. Compressez le contenu du dossier geometry dans un fichier .zip nommé geometry.zip. N'incluez pas le dossier geometry lui-même ; incluez uniquement le contenu du dossier comme illustré ci-après :

```
geometry.zip
   __init__.py
   trig/
      __init__.py
      line.py
```

2. Téléchargez geometry.zip dans votre compartiment Amazon S3.

> **Important**
>
> Si le compartiment Amazon S3 ne réside pas dans la même région que votre cluster Amazon Redshift, vous devez utiliser l'option REGION pour spécifier la région dans laquelle les données se trouvent. Pour plus d'informations, consultez CREATE LIBRARY.

3. A partir de votre outil client SQL, exécutez la commande suivante pour installer la bibliothèque. *<bucket_name>*Remplacez-le par le nom de votre compartiment, puis par *<access key id>* une clé *<secret key>* d'accès et une clé d'accès secrète provenant de vos informations d'identification utilisateur AWS Identity and Access Management (IAM).

```
CREATE LIBRARY geometry LANGUAGE plpythonu FROM 's3://<bucket_name>/geometry.zip'
 CREDENTIALS 'aws_access_key_id=<access key id>;aws_secret_access_key=<secret key>';
```

Après avoir installé la bibliothèque dans votre cluster, vous devez configurer vos fonctions en vue d'utiliser la bibliothèque. Pour ce faire, exécutez les commandes suivantes.

```
CREATE OR REPLACE FUNCTION f_distance (x1 float, y1 float, x2 float, y2 float) RETURNS
 float IMMUTABLE as $$
    from trig.line import LineSegment

    return LineSegment(x1, y1, x2, y2).distance()
$$ LANGUAGE plpythonu;

CREATE OR REPLACE FUNCTION f_within_range (x1 float, y1 float, x2 float, y2 float)
 RETURNS bool IMMUTABLE as $$
    from trig.line import LineSegment

    return LineSegment(x1, y1, x2, y2).distance() < 20
$$ LANGUAGE plpythonu;
```

Dans les commandes précédentes, `import trig/line` supprime le code dupliqué des fonctions d'origine dans cette section. Vous pouvez réutiliser les fonctionnalités fournies par cette bibliothèque à plusieurs reprises UDFs. Notez que pour importer le module, vous ne devez pas spécifier le chemin d'accès au nom du sous-package et du module (`trig/line`).

Contraintes UDF en Python

Dans le cadre des contraintes répertoriées dans cette rubrique, vous pouvez les utiliser UDFs partout où vous utilisez les fonctions scalaires intégrées d'Amazon Redshift. Pour de plus amples informations, veuillez consulter Référence sur les fonctions SQL.

Amazon Redshift Python est soumis aux UDFs contraintes suivantes :

- Python UDFs ne peut pas accéder au réseau ni lire ou écrire dans le système de fichiers.
- La taille totale des bibliothèques Python installées par l'utilisateur ne peut pas dépasser 100 Mo.
- Amazon Redshift ne peut exécuter qu'un seul UDF Python à la fois pour les clusters provisionnés à l'aide de la gestion automatique de la charge de travail (WLM) et pour les groupes de travail sans serveur. Si vous essayez d'exécuter plusieurs UDF simultanément, Amazon Redshift met en file d'attente le UDFs Python restant pour qu'il soit exécuté dans les files d'attente de gestion de la charge de travail. SQL UDFs n'a pas de limite de simultanéité lors de l'utilisation du WLM automatique.
- Lorsque vous utilisez le WLM manuel pour des clusters provisionnés, le nombre de Python UDFs pouvant être exécutés simultanément par cluster est limité à un quart du niveau de simultanéité

total du cluster. Par exemple, un cluster provisionné avec une simultanéité de 15 peut exécuter un maximum de trois Python simultanés. UDFs

- Lorsque vous utilisez Python UDFs, Amazon Redshift ne prend pas en charge les types de données SUPER et HLLSKETCH.

Erreurs de journalisation et avertissements en Python UDFs

Vous pouvez utiliser le module de journalisation Python pour créer des messages d'erreur et d'avertissement définis par l'utilisateur dans votre UDFs. Après l'exécution d'une requête, vous pouvez interroger la vue système SVL_UDF_LOG pour récupérer les messages journalisés.

> **Note**
>
> La journalisation UDF consomme des ressources de cluster et peut affecter les performances du système. Nous vous recommandons d'implémenter la journalisation uniquement pour le développement et le dépannage.

Lors de l'exécution de requête, le gestionnaire de journal écrit des messages dans la vue système SVL_UDF_LOG avec le noms de fonction, le nœud et la tranche correspondants. Le gestionnaire de journal écrit une ligne dans SVL_UDF_LOG par message, par tranche. Les messages sont tronqués à 4096 octets. Le journal UDF est limité à 500 lignes par tranche. Lorsque le journal est plein, le gestionnaire de journal élimine les messages les plus anciens et ajoute un message d'avertissement dans SVL_UDF_LOG.

> **Note**
>
> Le gestionnaire de journal UDF Amazon Redshift met en échappement les nouvelles lignes (\n), les barres verticale (|) et les barres obliques inverses (\ ) avec une barre oblique inverse (\ ).

Par défaut, la valeur de journal UDF est définie sur WARNING (Avertissement). Les messages avec le niveau de journal WARNING, ERROR et CRITICAL sont journalisés. Les messages avec une gravité plus faible INFO, DEBUG et NOTSET sont ignorés. Pour définir le niveau de journal UDF log level, utilisez la méthode logger Python. Par exemple, la commande suivante définit le niveau de journal sur INFO.

```
logger.setLevel(logging.INFO)
```

Pour plus d'informations sur l'utilisation du module de journalisation Python, consultez Logging facility for Python dans la documentation Python.

L'exemple suivant crée une fonction nommée f_pyerror qui importe le module de journalisation Python, instancie la méthode logger et journalise une erreur.

```
CREATE OR REPLACE FUNCTION f_pyerror()
RETURNS INTEGER
VOLATILE AS
$$
import logging

logger = logging.getLogger()
logger.setLevel(logging.INFO)
logger.info('Your info message here')
return 0
$$ language plpythonu;
```

L'exemple suivant interroge SVL_UDF_LOG pour afficher le message journalisé dans l'exemple précédent.

```
select funcname, node, slice, trim(message) as message
from svl_udf_log;

  funcname  | query | node | slice |   message
------------+-------+------+-------+-----------------
  f_pyerror | 12345 |     1|     1 | Your info message here
```

Lambda scalaire UDFs

Amazon Redshift peut utiliser des fonctions personnalisées définies dans le AWS Lambda cadre de requêtes SQL. Vous pouvez écrire le scalaire UDFs Lambda dans tous les langages de programmation pris en charge par Lambda, tels que Java, Go, Node.js PowerShell, C#, Python et Ruby. Vous pouvez également utiliser une exécution personnalisée.

La commande CREATE EXTERNAL FUNCTION crée les paramètres suivants :

- (Facultatif) Une liste d'arguments avec le type de données.

- Un type de données de retour.
- Un nom pour la fonction externe appelée par Amazon Redshift.
- Un rôle IAM que le cluster Amazon Redshift est autorisé à assumer pour effectuer un appel à Lambda.
- Un nom de fonction Lambda que la fonction UDF Lambda invoque.

Pour plus d'informations sur l'instruction CREATE EXTERNAL FUNCTION, consultez CREATE EXTERNAL FUNCTION.

Les types de données d'entrée et de retour pour Lambda UDFs peuvent être l'un des types suivants :

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISION
- CHAR
- VARCHAR
- BOOLEAN
- DATE
- TIMESTAMP

Les alias de ces types sont également valides. Pour une liste complète des types de données et de leurs alias, consultezTypes de données.

Amazon Redshift garantit que la fonction externe peut envoyer et recevoir des arguments et des résultats par lots.

Les Lambda UDFs sont définis et gérés dans Lambda, et vous pouvez contrôler les privilèges d'accès pour les invoquer dans Amazon UDFs Redshift. Vous pouvez invoquer plusieurs fonctions Lambda dans la même requête ou invoquer la même fonction plusieurs fois.

Utilisez Lambda UDFs dans toutes les clauses des instructions SQL où les fonctions scalaires sont prises en charge. Vous pouvez également utiliser Lambda UDFs dans n'importe quelle instruction SQL telle que SELECT, UPDATE, INSERT ou DELETE.

> **Note**
>
> L'utilisation de Lambda UDFs peut entraîner des frais supplémentaires liés au service Lambda. Cela dépend de facteurs tels que le nombre de requêtes Lambda (invocations UDF) et la durée totale de l'exécution du programme Lambda. Toutefois, l'utilisation de Lambda UDFs dans Amazon Redshift est gratuite. Pour plus d'informations sur la tarification AWS Lambda, consultez AWS Lambda la section Tarification.
>
> Le nombre de requêtes Lambda varie en fonction de la clause d'instruction SQL spécifique dans laquelle la fonction UDF Lambda est utilisée. Par exemple, supposons que la fonction soit utilisée dans une clause WHERE telle que la suivante.
>
> ```
> SELECT a, b FROM t1 WHERE lambda_multiply(a, b) = 64; SELECT a, b
> FROM t1 WHERE a*b = lambda_multiply(2, 32)
> ```
>
> Dans ce cas, Amazon Redshift appelle la première instruction SELECT pour chaque ligne et n'appelle la seconde instruction SELECT qu'une seule fois.
>
> Toutefois, l'utilisation d'une fonction UDF dans la partie projection de la requête peut n'invoquer la fonction Lambda qu'une seule fois pour chaque ligne qualifiée ou agrégée de l'ensemble de résultats.

Sécurité et autorisations UDF

Pour créer une fonction UDF Lambda, assurez-vous de disposer d'autorisations d'utilisation sur LANGUAGE EXFUNC. Vous devez accorder explicitement le privilège USAGE ON LANGUAGE EXFUNC, ou le révoquer, à des utilisateurs, des groupes ou des publics spécifiques.

L'exemple suivant accorde l'utilisation sur EXFUNC à PUBLIC.

```
grant usage on language exfunc to PUBLIC;
```

L'exemple suivant révoque le privilège USAGE pour exfunc de PUBLIC, puis accorde USAGE au groupe d'utilisateurs lambda_udf_devs.

```
revoke usage on language exfunc from PUBLIC;
```

```
grant usage on language exfunc to group lambda_udf_devs;
```

Pour exécuter une fonction UDF Lambda, assurez-vous que vous avez l'autorisation pour chaque fonction appelée. Par défaut, l'autorisation d'exécuter le nouveau Lambda UDFs est accordée à PUBLIC. Pour limiter l'utilisation, révoquez cette autorisation de PUBLIC pour la fonction. Ensuite, accordez le privilège à des utilisateurs ou à des groupes spécifiques.

L'exemple suivant révoque le privilège d'exécution de la fonction exfunc_sum de PUBLIC. Ensuite, il accorde le privilège USAGE au groupe d'utilisateurs lambda_udf_devs.

```
revoke execute on function exfunc_sum(int, int) from PUBLIC;
grant execute on function exfunc_sum(int, int) to group lambda_udf_devs;
```

Les super-utilisateurs ont tous les privilèges par défaut.

Pour plus d'informations sur l'octroi et la révocation des privilèges, consultez GRANT et REVOKE.

Configuration du paramètre d'autorisation pour Lambda UDFs

L'instruction CREATE EXTERNAL FUNCTION nécessite une autorisation pour invoquer les fonctions Lambda dans AWS Lambda. Pour démarrer l'autorisation, spécifiez un rôle AWS Identity and Access Management (IAM) lorsque vous exécutez la commande CREATE EXTERNAL FUNCTION. Pour plus d'informations sur les rôles IAM, consultez Rôles IAM dans le Guide de l'utilisateur IAM.

S'il existe un rôle IAM disposant des autorisations nécessaires pour invoquer les fonctions Lambda attachées à votre cluster, vous pouvez utiliser votre rôle Amazon Resource Name (ARN) dans le paramètre IAM_ROLE de l'instruction. Les sections suivantes décrivent les étapes à suivre pour utiliser un rôle IAM dans l'instruction CREATE EXTERNAL FUNCTION.

Création d'un rôle IAM pour Lambda

Le rôle IAM nécessite une autorisation pour invoquer des fonctions Lambda. Lorsque vous créez le rôle IAM, fournissez l'autorisation de l'une des manières suivantes :

- Attacher la politique `AWSLambdaRole` sur la page Attach permission policy (Attacher une politique d'autorisations) lorsque vous créez un rôle IAM. La politique `AWSLambdaRole` accorde des autorisations pour invoquer des fonctions Lambda, ce qui est la condition minimale. Pour plus d'informations et pour connaître les autres politiques, consultez la rubrique Stratégies IAM basées sur l'identité pour AWS Lambda dans le Manuel du développeur AWS Lambda .

- Créez votre propre politique personnalisée à joindre à votre rôle IAM avec l'autorisation `lambda:InvokeFunction` de toutes les ressources ou d'une fonction Lambda particulière avec l'ARN de cette fonction. Pour plus d'informations sur la création de politiques IAM, consultez la section Création de stratégies dans le Guide de l'utilisateur IAM.

  L'exemple de politique suivant permet d'invoquer Lambda sur une fonction Lambda particulière.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Invoke",
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": "arn:aws:lambda:us-west-2:123456789012:function:my-function"
        }
    ]
}
```

  Pour plus d'informations sur les ressources pour les fonctions Lambda, consultez la rubrique Ressources et conditions pour les actions Lambda dans la Référence API IAM.

  Après avoir créé votre politique personnalisée avec les autorisations requises, vous pouvez l'attacher au rôle IAM sur la page Attacher une politique d'autorisations lorsque vous créez un rôle IAM.

Pour savoir comment créer un rôle IAM, consultez la section Autoriser Amazon Redshift à accéder à AWS d'autres services en votre nom dans le guide de gestion Amazon Redshift.

Si vous ne souhaitez pas créer un nouveau rôle IAM, vous pouvez ajouter les autorisations mentionnées précédemment à votre rôle IAM existant.

Association d'un rôle IAM au cluster

Attachez le rôle IAM à votre cluster. Vous pouvez ajouter un rôle à un cluster ou afficher les rôles associés à un cluster à l'aide de la console de gestion, de la CLI ou de l'API Amazon Redshift. Pour plus d'informations, consultez la rubrique Associer un rôle IAM à un cluster dans le Guide de gestion Amazon Redshift.

Inclusion du rôle IAM dans l'instruction

Incluez le rôle IAM ARN dans l'instruction CREATE EXTERNAL FUNCTION. Lorsque vous créez un rôle IAM, IAM renvoie un Amazon Resource Name (ARN) pour le rôle. Pour spécifier un rôle IAM, indiquez l'ARN de rôle avec le paramètre `IAM_ROLE`. L'exemple suivant montre la syntaxe du paramètre `IAM_ROLE`.

```
IAM_ROLE 'arn:aws:iam::aws-account-id:role/role-name'
```

Pour invoquer des fonctions Lambda qui résident dans d'autres comptes au sein de la même région, consultez Chaîner les rôles IAM dans Amazon Redshift.

Utilisation de l'interface JSON entre Amazon Redshift et AWS Lambda

Amazon Redshift utilise une interface commune pour toutes les fonctions Lambda avec lesquelles Amazon Redshift communique.

Le tableau suivant présente la liste des champs d'entrée des fonctions Lambda désignées auxquels vous pouvez vous attendre pour la charge utile JSON.

Nom de champ	Description	Plage de valeurs
request_id	Identifiant universel unique (UUID) qui identifie de manière unique chaque requête d'invocation.	Un UUID valide.
cluster	Amazon Resource Name (ARN) complet du cluster.	Un ARN de cluster valide.
utilisateur	Nom de l'utilisateur qui effectue l'appel.	Un nom d'utilisateur valide.
database	Nom de la base de données sur laquelle la requête est exécutée.	Un nom de base de données valide.

Nom de champ	Description	Plage de valeurs
external_function	Nom entièrement qualifié de la fonction externe qui effectue l'appel.	Un nom de fonction complet valide.
query_id	ID de la requête qui effectue l'appel.	Un ID de requête valide.
num_records	Nombre d'arguments dans la charge utile.	Une valeur de 1 à 2^64.
arguments	Charge utile de données dans le format spécifié.	Les données au format tableau doivent être un tableau JSON. Chaque élément est un enregistrement qui est un tableau si le nombre d'arguments est supérieur à 1. En utilisant un tableau, Amazon Redshift préserve l'ordre des enregistrements dans la charge utile.

L'ordre du tableau JSON détermine l'ordre du traitement par lots. La fonction Lambda doit traiter les arguments de manière itérative et produire le nombre exact d'enregistrements. Voici un exemple de charge utile.

```
{
  "request_id" : "23FF1F97-F28A-44AA-AB67-266ED976BF40",
  "cluster" : "arn:aws:redshift:xxxx",
  "user" : "adminuser",
  "database" : "db1",
  "external_function": "public.foo",
  "query_id" : 5678234,
  "num_records" : 4,
  "arguments" : [
     [ 1, 2 ],
     [ 3, null],
     null,
     [ 4, 6]
   ]
 }
```

La sortie de retour de la fonction Lambda contient les champs suivants.

Nom de champ	Description	Plage de valeurs
success	Indication du succès ou de l'échec de la fonction.	Une valeur de `"true"` ou `"false"`.
error_msg	Message d'erreur si la valeur de succès est `"false"` (si la fonction échoue) ; sinon, ce champ est ignoré.	Un message valide.
num_records	Nombre d'enregis trements dans la charge utile.	Une valeur de 1 à 2^64.
results	Résultats de l'appel dans le format spécifié.	N/A

Voici un exemple de sortie de fonction Lambda.

```
{
  "success": true,   // true indicates the call succeeded
  "error_msg" : "my function isn't working",  // shall only exist when success != true
  "num_records": 4,      // number of records in this payload
  "results" : [
     1,
     4,
     null,
     7
   ]
}
```

Lorsque vous appelez des fonctions Lambda à partir de requêtes SQL, Amazon Redshift assure la sécurité de la connexion avec les considérations suivantes :

- Privilèges GRANT et REVOKE. Pour plus d'informations sur la sécurité et les autorisations UDF, consultezSécurité et autorisations UDF.
- Amazon Redshift soumet uniquement le jeu minimal de données à la fonction Lambda désignée.
- Amazon Redshift appelle uniquement la fonction Lambda désignée avec le rôle IAM désigné.

Exemples de cas d'utilisation pour les fonctions définies par l'utilisateur () UDFs

Vous pouvez utiliser des fonctions définies par l'utilisateur pour résoudre des problèmes commerciaux en intégrant Amazon Redshift à d'autres composants. Voici quelques exemples de la façon dont d'autres l'ont utilisé UDFs pour leurs cas d'utilisation :

- Accès aux composants externes à l'aide d'Amazon Redshift Lambda UDFs : décrit le UDFs fonctionnement d'Amazon Redshift Lambda et explique comment créer un UDF Lambda.
- Traduisez et analysez du texte à l'aide des fonctions SQL avec Amazon Redshift, Amazon Translate et Amazon Comprehend : fournit Amazon Redshift UDFs Lambda prédéfini que vous pouvez installer en quelques clics pour traduire, rédiger et analyser des champs de texte.
- Accédez à Amazon Location Service depuis Amazon Redshift : décrit comment utiliser Amazon Redshift UDFs Lambda pour l'intégrer à Amazon Location Service.
- Tokenisation des données avec Amazon Redshift et Protegrity : décrit comment intégrer Amazon Redshift Lambda au produit Protegrity Serverless. UDFs
- Amazon Redshift UDFs : collection d'Amazon Redshift SQL, Lambda et Python. UDFs

Création de procédures stockées dans Amazon Redshift

Cette rubrique explique comment créer et utiliser des procédures stockées dans Amazon Redshift. Une procédure stockée est un ensemble d'instructions SQL que plusieurs programmes peuvent utiliser.

Vous pouvez définir une procédure stockée Amazon Redshift à l'aide de PL/pgSQ, le langage procédural de PostgreSQL, pour exécuter un ensemble de requêtes SQL et d'opérations logiques. La procédure est stockée dans la base de données et est accessible à tout utilisateur disposant de privilèges suffisants dans la base de données.

Contrairement à une fonction définie par l'utilisateur, une procédure stockée peut incorporer des instructions DDL (Data-Definition Language) et DML (Data-Manipulation Language), en plus des requêtes SELECT. Une procédure stockée n'a pas besoin de retourner une valeur. Vous pouvez utiliser le langage procédural, y compris les boucles et les expressions conditionnelles, pour contrôler le flux logique.

Pour plus d'informations sur les commandes SQL qui permettent de créer et de gérer des procédures stockées, consultez les rubriques suivantes :

- CREATE PROCEDURE
- ALTER PROCEDURE
- DROP PROCEDURE
- SHOW PROCEDURE
- CALL
- GRANT
- REVOKE
- ALTER DEFAULT PRIVILEGES

Rubriques

- Présentation des procédures stockées dans Amazon Redshift
- Guide de référence du langage PL/pgSQL

Présentation des procédures stockées dans Amazon Redshift

Cette rubrique décrit les détails relatifs à l'objectif et à l'utilisation des procédures stockées.

Les procédures stockées permettent généralement d'encapsuler la logique de transformation et de validation des données, et la logique propre à l'activité. En associant plusieurs étapes SQL dans une même procédure stockée, vous pouvez réduire les allers-retours entre vos applications et la base de données.

Pour un contrôle d'accès détaillé, vous pouvez créer des procédures stockées qui permettent d'exécuter des fonctions sans offrir à l'utilisateur un accès aux tables sous-jacentes. Par exemple, seul le propriétaire ou un superutilisateur peut tronquer une table, et un utilisateur a besoin de privilèges d'écriture pour insérer des données dans une table. Au lieu d'accorder à un utilisateur des privilèges sur les tables sous-jacentes, vous pouvez créer une procédure stockée qui exécute la tâche. Vous accordez ensuite à l'utilisateur les privilèges nécessaires à l'exécution de la procédure stockée.

Une procédure stockée dotée de l'attribut de sécurité DEFINER s'exécute avec les privilèges du propriétaire de la procédure stockée. Par défaut, une procédure stockée dispose d'une sécurité INVOKER, ce qui signifie que la procédure utilise les privilèges de l'utilisateur qui l'appelle.

Pour créer une procédure stockée, utilisez la commande CREATE PROCEDURE. Pour exécuter une procédure, utilisez la commande CALL. Vous trouverez des exemples plus loin dans la présente section.

> **Note**
>
> Certains clients peuvent rencontrer l'erreur suivante lors de la création d'une procédure stockée Amazon Redshift.
>
> ```
> ERROR: 42601: [Amazon](500310) unterminated dollar-quoted string at or near "$$
> ```
>
> Cette erreur est due à l'incapacité du client à analyser correctement l'instruction CREATE PROCEDURE avec des points-virgules délimitant les instructions et le signe dollar ($) entre guillemets. Il en résulte qu'une partie seulement de l'instruction est envoyée au serveur Amazon Redshift. Généralement, vous pouvez contourner cette erreur à l'aide de l'option `Run as batch` ou `Execute selected` du client.
> Par exemple, avec un client Aginity, utilisez l'option `Run entire script as batch`.
> Lorsque vous utilisez la Workbench/J, we recommend version 124. When you use SQL

Workbench/J version 125 de SQL, pensez à spécifier un autre délimiteur pour contourner le problème.
CREATE PROCEDURE contient des instructions SQL délimitées par un point-virgule (;). La définition d'un autre délimiteur tel que la barre oblique (/) et son positionnement à la fin de l'instruction CREATE PROCEDURE permet que l'instruction soit envoyée au serveur Amazon Redshift pour traitement. Voici un exemple.

```
CREATE OR REPLACE PROCEDURE test()
AS $$
BEGIN
  SELECT 1 a;
END;
$$
LANGUAGE plpgsql
;
/
```

Vous pouvez utiliser un client prenant en charge l'analyse des instructions CREATE PROCEDURE, tel que l'éditeur de requêtes de la console Amazon Redshift ou. TablePlus

Rubriques

- Dénomination des procédures stockées
- Sécurité et privilèges des procédures stockées
- Renvoyer un jeu de résultats à partir d'une procédure stockée
- Gestion des transactions
- Interception des erreurs
- Enregistrement des procédures stockées
- Limites des procédures stockées

L'exemple suivant illustre une procédure sans arguments en sortie. Par défaut, les arguments sont des arguments en entrée (IN).

```
CREATE OR REPLACE PROCEDURE test_sp1(f1 int, f2 varchar)
AS $$
BEGIN
```

```
  RAISE INFO 'f1 = %, f2 = %', f1, f2;
END;
$$ LANGUAGE plpgsql;

call test_sp1(5, 'abc');
INFO: f1 = 5, f2 = abc
CALL
```

> **Note**
>
> Lorsque vous écrivez des procédures stockées, nous vous recommandons une bonne pratique pour sécuriser les valeurs sensibles :
> Ne codez pas en dur des informations sensibles dans la logique des procédures stockées. Par exemple, n'attribuez pas de mot de passe utilisateur dans une instruction CREATE USER dans le corps d'une procédure stockée. Cela présente un risque pour la sécurité, car les valeurs codées en dur peuvent être enregistrées en tant que métadonnées de schéma dans les tables du catalogue. Transmettez plutôt des valeurs sensibles, telles que des mots de passe, en tant qu'arguments à la procédure stockée, au moyen de paramètres.
> Pour plus d'informations sur les procédures stockées, consultez CREATE PROCEDURE et Création de procédures stockées dans Amazon Redshift. Pour plus d'informations sur les tables de catalogue, consultez Tables catalogue système.

L'exemple suivant illustre une procédure avec des arguments en sortie. Les arguments sont en entrée (IN), en entrée et en sortie (INOUT), et en sortie (OUT).

```
CREATE OR REPLACE PROCEDURE test_sp2(f1 IN int, f2 INOUT varchar(256), out_var OUT
 varchar(256))
AS $$
DECLARE
  loop_var int;
BEGIN
  IF f1 is null OR f2 is null THEN
    RAISE EXCEPTION 'input cannot be null';
  END IF;
  DROP TABLE if exists my_etl;
  CREATE TEMP TABLE my_etl(a int, b varchar);
    FOR loop_var IN 1..f1 LOOP
        insert into my_etl values (loop_var, f2);
```

```
        f2 := f2 || '+' || f2;
    END LOOP;
  SELECT INTO out_var count(*) from my_etl;
END;
$$ LANGUAGE plpgsql;


call test_sp2(2,'2019');

         f2          | column2
---------------------+---------
 2019+2019+2019+2019 | 2
(1 row)
```

Dénomination des procédures stockées

Cette rubrique décrit les détails relatifs aux noms de procédures stockées.

Si vous définissez une procédure avec le même nom, mais des types de données différents pour les arguments en entrée, vous créez une nouvelle procédure. Par conséquent, le nom de la procédure est surchargé. Pour de plus amples informations, veuillez consulter Surcharge des noms de procédure. Amazon Redshift ne permet pas de surcharger les procédures en fonction des arguments de sortie. Vous ne pouvez pas avoir deux procédures ayant le même nom et les mêmes types de données pour les arguments en entrée, mais des types différents pour les arguments en sortie.

Le propriétaire ou un super-utilisateur peut remplacer le corps d'une procédure stockée par une nouvelle procédure ayant la même signature. Pour modifier la signature ou les types de retour d'une procédure stockée, supprimez la procédure stockée et recréez-la. Pour plus d'informations, consultez DROP PROCEDURE et CREATE PROCEDURE.

Vous pouvez éviter les conflits potentiels et les résultats inattendus en prenant en considération vos conventions de dénomination des procédures stockées avant de les mettre en œuvre. Comme vous pouvez surcharger les noms de procédure, ceux-ci peuvent entrer en conflit avec des noms de procédure Amazon Redshift existants et futurs.

Surcharge des noms de procédure

Une procédure est identifiée par son nom et sa signature, à savoir le nombre d'arguments en entrée et leurs types de données. Deux procédures d'un même schéma peuvent porter le même nom si elles ont des signatures différentes. Autrement dit, vous pouvez surcharger les noms de procédure.

Lorsque vous exécutez une procédure, le moteur de requête détermine quelle procédure appeler en fonction du nombre d'arguments que vous fournissez et de leurs types de données. Vous pouvez utiliser une surcharge pour simuler des procédures ayant un nombre variable d'arguments, jusqu'à la limite autorisée par la commande CREATE PROCEDURE. Pour de plus amples informations, veuillez consulter CREATE PROCEDURE.

Prévention des conflits de dénomination

Nous vous recommandons de nommer toutes les procédures en utilisant le préfixe `sp_`. Amazon Redshift réserve le préfixe `sp_` exclusivement aux procédures stockées. En préfixant vos noms de procédure avec `sp_`, vous garantissez que le nom de votre procédure n'entrera pas en conflit avec le nom de procédure Amazon Redshift existant ou futur.

Sécurité et privilèges des procédures stockées

Cette rubrique décrit les informations d'identification de base de données nécessaires pour créer et exécuter des procédures stockées.

Par défaut, tous les utilisateurs disposent des privilèges nécessaires pour créer une procédure. Pour créer une procédure, vous devez disposer du privilège USAGE sur la langue PL/pgSQL, which is granted to PUBLIC by default. Only superusers and owners have the privilege to call a procedure by default. Superusers can run REVOKE USAGE on PL/pgSQL d'un utilisateur s'il souhaite empêcher l'utilisateur de créer une procédure stockée.

Pour appeler une procédure, vous devez disposer du privilège EXECUTE sur la procédure. Par défaut, le privilège EXECUTE pour les nouvelles procédures est accordé au propriétaire de la procédure et aux super-utilisateurs. Pour de plus amples informations, veuillez consulter GRANT.

L'utilisateur qui crée une procédure est le propriétaire par défaut. Par défaut, le propriétaire possède les privilèges CREATE, DROP et EXECUTE sur la procédure. Les super-utilisateurs disposent de tous les privilèges.

L'attribut SECURITY contrôle les privilèges d'une procédure relatifs à l'accès aux objets de base de données. Lorsque vous créez une procédure stockée, vous pouvez définir l'attribut SECURITY sur DEFINER ou INVOKER. Cet attribut détermine les privilèges utilisés lors de l'exécution des instructions dans le corps de la procédure stockée. Si vous spécifiez SECURITY INVOKER, la procédure utilise les privilèges de l'utilisateur invoquant la procédure. Si vous spécifiez SECURITY DEFINER, la procédure utilise les privilèges du propriétaire de la procédure. INVOKER est la valeur par défaut.

Parce qu'une procédure SECURITY DEFINER s'exécute avec les privilèges de l'utilisateur qui la possède, vous devez vous assurer que la procédure ne peut pas être utilisée à mauvais escient. Pour vous assurer que les procédures SECURITY DEFINER ne peuvent pas être utilisées à mauvais escient, procédez comme suit :

- Accordez l'autorisation EXECUTE sur les procédures définies avec SECURITY DEFINER à des utilisateurs spécifiques, et non à PUBLIC.
- Qualifiez tous les objets de base de données dont la procédure a besoin pour accéder à l'aide des noms de schéma. Par exemple, utilisez `myschema.mytable` plutôt que `mytable`.
- Si vous ne pouvez pas qualifier un nom d'objet par son schéma, lors de la création de la procédure, définissez `search_path` à l'aide de l'option SET. Définissez `search_path` de façon à exclure tout schéma accessible en écriture par des utilisateurs non fiables. Cette approche empêche tout appelant de la procédure de créer des objets (tels que des tables ou des vues) qui masquent les objets destinés à être utilisés par la procédure. Pour plus d'informations sur l'option SET, consultez CREATE PROCEDURE.

L'exemple suivant définit `search_path` sur `admin` pour s'assurer que la table `user_creds` est accessible depuis le schéma `admin` et non depuis le schéma public ou autre défini dans le `search_path` de l'appelant.

```
CREATE OR REPLACE PROCEDURE sp_get_credentials(userid int, o_creds OUT varchar)
AS $$
BEGIN
  SELECT creds INTO o_creds
  FROM user_creds
  WHERE user_id = $1;
END;
$$ LANGUAGE plpgsql
SECURITY DEFINER
-- Set a secure search_path
SET search_path = admin;
```

Renvoyer un jeu de résultats à partir d'une procédure stockée

Cette rubrique décrit la manière dont les procédures stockées renvoient des données.

Vous pouvez retourner un ensemble de résultats à l'aide d'un curseur ou d'une table temporaire.

Retour d'un curseur

Pour retourner un curseur, créez une procédure avec un argument INOUT défini avec un type de données `refcursor`. Lorsque vous appelez la procédure, donnez un nom au curseur. Vous pouvez ensuite extraire les résultats du curseur par leur nom.

L'exemple suivant crée une procédure nommée `get_result_set` avec un argument INOUT nommé `rs_out` à l'aide du type de données `refcursor`. La procédure ouvre le curseur à l'aide d'une instruction SELECT.

```
CREATE OR REPLACE PROCEDURE get_result_set (param IN integer, rs_out INOUT refcursor)
AS $$
BEGIN
  OPEN rs_out FOR SELECT * FROM fact_tbl where id >= param;
END;
$$ LANGUAGE plpgsql;
```

La commande CALL suivante ouvre le curseur avec le nom `mycursor`. N'utlisez les curseurs qu'au sein des transactions.

```
BEGIN;
CALL get_result_set(1, 'mycursor');
```

Une fois que le curseur est ouvert, vous pouvez procéder à l'extraction depuis ce curseur, comme l'illustre l'exemple suivant.

```
FETCH ALL FROM mycursor;

    id | secondary_id | name
-------+--------------+---------
     1 |            1 | Joe
     1 |            2 | Ed
     2 |            1 | Mary
     1 |            3 | Mike
(4 rows)
```

À la fin, la transaction est validée ou annulée.

```
COMMIT;
```

Un curseur retourné par une procédure stockée est soumise aux mêmes contraintes et considérations de performance, telles que décrites dans DECLARE CURSOR; Pour de plus amples informations, veuillez consulter Contraintes de curseur.

L'exemple suivant montre l'appel de la procédure stockée `get_result_set` à l'aide d'un type de données `refcursor` depuis JDBC. Le littéral `'mycursor'` (nom du curseur) est transmis à `prepareStatement`. Puis, les résultats sont extraits de `ResultSet`.

```
static void refcursor_example(Connection conn) throws SQLException {
    conn.setAutoCommit(false);
    PreparedStatement proc = conn.prepareStatement("CALL get_result_set(1,
 'mycursor')");
    proc.execute();
    ResultSet rs = statement.executeQuery("fetch all from mycursor");
    while (rs.next()) {
      int n = rs.getInt(1);
      System.out.println("n " + n);
    }
```

Utilisation d'une table temporaire

Pour retourner les résultats, vous pouvez retourner un descripteur vers une table temporaire contenant les lignes de résultats. Le client peut fournir un nom comma paramètre à la procédure stockée. À l'intérieur de la procédure stockée, Dynamic SQL peut être utilisé pour œuvrer sur la table temporaire. Vous en trouverez un exemple ci-dessous.

```
CREATE PROCEDURE get_result_set(param IN integer, tmp_name INOUT varchar(256)) as $$
DECLARE
  row record;
BEGIN
  EXECUTE 'drop table if exists ' || tmp_name;
  EXECUTE 'create temp table ' || tmp_name || ' as select * from fact_tbl where id <= '
 || param;
END;
$$ LANGUAGE plpgsql;

CALL get_result_set(2, 'myresult');
 tmp_name
-----------
 myresult
(1 row)
```

```
SELECT * from myresult;
 id | secondary_id | name
----+--------------+------
  1 |            1 | Joe
  2 |            1 | Mary
  1 |            2 | Ed
  1 |            3 | Mike
(4 rows)
```

Gestion des transactions

Vous pouvez créer une procédure stockée avec un comportement de gestion des transactions par défaut ou un comportement non atomique.

Mode par défaut de gestion des transactions des procédures stockées

Le comportement de validation automatique du mode transactionnel par défaut entraîne la validation individuelle de chaque commande SQL qui s'exécute séparément. Un appel à une procédure stockée est traité comme une simple commande SQL. Les instructions SQL à l'intérieur d'une procédure se comportent comme si elles étaient dans un bloc de transaction qui commence implicitement quand l'appel démarre et se termine quand l'appel finit. Un appel imbriqué à une autre procédure est traité comme toute autre instruction SQL et opère au sein du contexte de la même transaction que l'appelant. Pour plus d'informations sur le comportement automatique de validation, consultez Isolement sérialisable.

Cependant, supposons que vous appelez une procédure stockée depuis un bloc de transaction spécifié par l'utilisateur (défini par BEGIN...COMMIT). Dans ce cas, toutes les instructions de la procédure stockée s'exécutent dans le contexte de la transaction spécifiée par l'utilisateur. La procédure n'est pas validée implicitement à la sortie. L'appelant contrôle la validation ou l'annulation de la procédure.

En cas d'erreur lors de l'exécution d'une procédure stockée, toutes les modifications apportées lors de la transaction en cours sont annulées.

Vous pouvez utiliser les instructions de contrôle de transaction suivantes dans une procédure stockée :

- COMMIT : valide l'ensemble du travail effectué dans la transaction actuelle et démarre implicitement une nouvelle transaction. Pour de plus amples informations, veuillez consulter COMMIT.

- ROLLBACK : restaure le travail effectué dans la transaction actuelle et démarre implicitement une nouvelle transaction. Pour de plus amples informations, veuillez consulter ROLLBACK.

TRUNCATE est une autre instruction qui peut être émise depuis une procédure stockée et qui influe sur la gestion de la transaction. Dans Amazon Redshift, TRUNCATE émet une validation implicitement. Ce comportement demeure le même dans le contexte des procédures stockées. Quand une instruction TRUNCATE est émise depuis une procédure stockée, elle valide la transaction en cours et en démarre une nouvelle. Pour de plus amples informations, veuillez consulter TRUNCATE.

Toutes les instructions qui suivent une instruction COMMIT, ROLLBACK ou TRUNCATE s'exécutent dans le contexte d'une nouvelle transaction. C'est le cas jusqu'à ce qu'une instruction COMMIT, ROLLBACK ou TRUNCATE soit détectée ou que la procédure stockée se termine.

Lorsque vous utilisez une instruction COMMIT, ROLLBACK ou TRUNCATE depuis une procédure stockée, les contraintes suivantes s'appliquent :

- Si la procédure stockée est appelée depuis un bloc de transaction, elle ne peut pas émettre d'instruction COMMIT, ROLLBACK ou TRUNCATE. Cette restriction s'applique dans le corps de la procédure stockée elle-même et dans tout appel de procédure imbriquée.
- Si la procédure stockée est créée avec les options `SET config`, elle ne peut pas émettre d'instruction COMMIT, ROLLBACK ou TRUNCATE. Cette restriction s'applique dans le corps de la procédure stockée elle-même et dans tout appel de procédure imbriquée.
- Tout curseur ouvert, explicitement ou implicitement, est fermé automatiquement quand une instruction COMMIT, ROLLBACK ou TRUNCATE est traitée. Pour les contraintes sur les curseurs explicites et implicites, consultez Limites des procédures stockées.

Vous ne pouvez pas exécuter COMMIT ou ROLLBACK avec Dynamic SQL. Toutefois, TRUNCATE peut être exécuté avec Dynamic SQL. Pour de plus amples informations, veuillez consulter Instructions SQL dynamiques.

Lorsque vous travaillez avec des procédures stockées, considérez que les instructions BEGIN et END dans PL/pgSQL sont destinées uniquement au regroupement. Ils ne démarrent pas et ne finissent pas une transaction. Pour de plus amples informations, veuillez consulter Bloc.

L'exemple suivant illustre le comportement d'une transaction lors de l'appel d'une procédure stockée depuis un bloc de transaction explicite. Les deux instructions d'insertion émises depuis l'extérieur de

la procédure stockée et celle émise depuis l'intérieur font toutes deux partie de la même transaction (3382). La transaction est validée quand l'utilisateur émet la validation explicite.

```
CREATE OR REPLACE PROCEDURE sp_insert_table_a(a int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO test_table_a values (a);
END;
$$;

Begin;
  insert into test_table_a values (1);
  Call sp_insert_table_a(2);
  insert into test_table_a values (3);
Commit;

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  | pid |  type   |              stmt_text
--------+------+-----+---------+---------------------------------------
    103 | 3382 | 599 | UTILITY | Begin;
    103 | 3382 | 599 | QUERY   | insert into test_table_a values (1);
    103 | 3382 | 599 | UTILITY | Call sp_insert_table_a(2);
    103 | 3382 | 599 | QUERY   | INSERT INTO test_table_a values ( $1 )
    103 | 3382 | 599 | QUERY   | insert into test_table_a values (3);
    103 | 3382 | 599 | UTILITY | COMMIT
```

À l'inverse, prenons un exemple où les mêmes instructions sont émises depuis l'extérieur d'un bloc de transactions explicite et où le paramètre de validation automatique de la session est activé. Dans ce cas, chaque instruction s'exécute dans sa propre transaction.

```
insert into test_table_a values (1);
Call sp_insert_table_a(2);
insert into test_table_a values (3);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  | pid |  type   |
             stmt_text
```

```
--------+------+-----+---------
+-----------------------------------------------------------------------------------------------
    103 | 3388 | 599 | QUERY   | insert into test_table_a values (1);
    103 | 3388 | 599 | UTILITY | COMMIT
    103 | 3389 | 599 | UTILITY | Call sp_insert_table_a(2);
    103 | 3389 | 599 | QUERY   | INSERT INTO test_table_a values ( $1 )
    103 | 3389 | 599 | UTILITY | COMMIT
    103 | 3390 | 599 | QUERY   | insert into test_table_a values (3);
    103 | 3390 | 599 | UTILITY | COMMIT
```

L'exemple suivant émet une instruction TRUNCATE après l'insertion dans `test_table_a`. L'instruction TRUNCATE émet une validation implicite qui valide la transaction courante (3335) et en démarre une nouvelle (3336). La nouvelle transaction est validée quand la procédure cesse.

```
CREATE OR REPLACE PROCEDURE sp_truncate_proc(a int, b int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO test_table_a values (a);
  TRUNCATE test_table_b;
  INSERT INTO test_table_b values (b);
END;
$$;

Call sp_truncate_proc(1,2);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |  pid  |  type   |
                                        stmt_text
--------+------+-------+---------
+-----------------------------------------------------------------------------------------------
    103 | 3335 | 23636 | UTILITY | Call sp_truncate_proc(1,2);
    103 | 3335 | 23636 | QUERY   | INSERT INTO test_table_a values ( $1 )
    103 | 3335 | 23636 | UTILITY | TRUNCATE test_table_b
    103 | 3335 | 23636 | UTILITY | COMMIT
    103 | 3336 | 23636 | QUERY   | INSERT INTO test_table_b values ( $1 )
    103 | 3336 | 23636 | UTILITY | COMMIT
```

L'exemple suivant émet une commande TRUNCATE depuis un appel imbriqué. L'instruction TRUNCATE valide tout le travail effectué jusque-là dans les procédures externes et internes d'une

transaction (3344). Elle démarre une nouvelle transaction (3345). La nouvelle transaction est validée quand la procédure externe cesse.

```
CREATE OR REPLACE PROCEDURE sp_inner(c int, d int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO inner_table values (c);
  TRUNCATE outer_table;
  INSERT INTO inner_table values (d);
END;
$$;

CREATE OR REPLACE PROCEDURE sp_outer(a int, b int, c int, d int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO outer_table values (a);
  Call sp_inner(c, d);
  INSERT INTO outer_table values (b);
END;
$$;

Call sp_outer(1, 2, 3, 4);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |  pid  |  type   |
                                        stmt_text
--------+------+-------+---------
+-----------------------------------------------------------------------------------------------
    103 | 3344 | 23636 | UTILITY | Call sp_outer(1, 2, 3, 4);
    103 | 3344 | 23636 | QUERY   | INSERT INTO outer_table values ( $1 )
    103 | 3344 | 23636 | UTILITY | CALL sp_inner( $1 , $2 )
    103 | 3344 | 23636 | QUERY   | INSERT INTO inner_table values ( $1 )
    103 | 3344 | 23636 | UTILITY | TRUNCATE outer_table
    103 | 3344 | 23636 | UTILITY | COMMIT
    103 | 3345 | 23636 | QUERY   | INSERT INTO inner_table values ( $1 )
    103 | 3345 | 23636 | QUERY   | INSERT INTO outer_table values ( $1 )
    103 | 3345 | 23636 | UTILITY | COMMIT
```

L'exemple suivant montre que le curseur `cur1` a été fermé quand l'instruction TRUNCATE a été validée.

```
CREATE OR REPLACE PROCEDURE sp_open_cursor_truncate()
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  cur1 cursor for select * from test_table_a order by 1;
BEGIN
  open cur1;
  TRUNCATE table test_table_b;
  Loop
    fetch cur1 into rec;
    raise info '%', rec.c1;
    exit when not found;
  End Loop;
END
$$;

call sp_open_cursor_truncate();
ERROR: cursor "cur1" does not exist
CONTEXT: PL/pgSQL function "sp_open_cursor_truncate" line 8 at fetch
```

L'exemple suivant émet une instruction TRUNCATE et ne peut pas être appelé depuis un bloc de transaction explicite.

```
CREATE OR REPLACE PROCEDURE sp_truncate_atomic() LANGUAGE plpgsql
AS $$
BEGIN
  TRUNCATE test_table_b;
END;
$$;

Begin;
  Call sp_truncate_atomic();
ERROR: TRUNCATE cannot be invoked from a procedure that is executing in an atomic
 context.
HINT: Try calling the procedure as a top-level call i.e. not from within an explicit
 transaction block.
Or, if this procedure (or one of its ancestors in the call chain) was created with SET
 config options, recreate the procedure without them.
CONTEXT: SQL statement "TRUNCATE test_table_b"
PL/pgSQL function "sp_truncate_atomic" line 2 at SQL statement
```

L'exemple suivant montre qu'un utilisateur qui n'est pas un super-utilisateur ni le propriétaire d'une table peut émettre une instruction TRUNCATE sur la table. L'utilisateur effectue cette opération à l'aide d'une procédure stockée `Security Definer`. L'exemple illustre les actions suivantes :

- L'utilisateur1 crée la table `test_tbl`.
- L'utilisateur1 crée une procédure stockée `sp_truncate_test_tbl`.
- L'utilisateur1 accorde un privilège EXECUTE au niveau de la procédure stockée à l'utilisateur2.
- L'utilisateur2 exécute la procédure stockée pour tronquer la table `test_tbl`. L'exemple montre le nombre de lignes avant et après la commande TRUNCATE.

```
set session_authorization to user1;
create table test_tbl(id int, name varchar(20));
insert into test_tbl values (1,'john'), (2, 'mary');
CREATE OR REPLACE PROCEDURE sp_truncate_test_tbl() LANGUAGE plpgsql
AS $$
DECLARE
  tbl_rows int;
BEGIN
  select count(*) into tbl_rows from test_tbl;
  RAISE INFO 'RowCount before Truncate: %', tbl_rows;
  TRUNCATE test_tbl;
  select count(*) into tbl_rows from test_tbl;
  RAISE INFO 'RowCount after Truncate: %', tbl_rows;
END;
$$ SECURITY DEFINER;
grant execute on procedure sp_truncate_test_tbl() to user2;
reset session_authorization;


set session_authorization to user2;
call sp_truncate_test_tbl();
INFO:  RowCount before Truncate: 2
INFO:  RowCount after Truncate: 0
CALL
reset session_authorization;
```

L'exemple suivant émet deux fois COMMIT. Le premier COMMIT valide l'ensemble du travail effectué dans la transaction 10363 et démarre implicitement la transaction 10364. La transaction 10364 est validée par la deuxième instruction COMMIT.

```
CREATE OR REPLACE PROCEDURE sp_commit(a int, b int) LANGUAGE plpgsql
AS $$
BEGIN
  INSERT INTO test_table values (a);
  COMMIT;
  INSERT INTO test_table values (b);
  COMMIT;
END;
$$;

call sp_commit(1,2);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;
 userid |  xid  | pid  |  type   |
                               stmt_text
--------+-------+------+---------
+-----------------------------------------------------------------------------------------------------
    100 | 10363 | 3089 | UTILITY | call sp_commit(1,2);
    100 | 10363 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10363 | 3089 | UTILITY | COMMIT
    100 | 10364 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10364 | 3089 | UTILITY | COMMIT
```

L'exemple suivant émet une instruction ROLLBACK si `sum_vals` est supérieur à 2. La première instruction ROLLBACK restaure tout le travail effectué dans la transaction 10377 et démarre une nouvelle transaction 10378. La transaction 10378 est validée quand la procédure se termine.

```
CREATE OR REPLACE PROCEDURE sp_rollback(a int, b int) LANGUAGE plpgsql
AS $$
DECLARE
  sum_vals int;
BEGIN
  INSERT INTO test_table values (a);
  SELECT sum(c1) into sum_vals from test_table;
  IF sum_vals > 2 THEN
    ROLLBACK;
  END IF;

  INSERT INTO test_table values (b);
END;
```

```
$$;

call sp_rollback(1, 2);

select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

userid |  xid  | pid  |  type   |
                              stmt_text
--------+-------+------+---------
+-----------------------------------------------------------------------------------------------------
    100 | 10377 | 3089 | UTILITY | call sp_rollback(1, 2);
    100 | 10377 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10377 | 3089 | QUERY   | SELECT sum(c1) from test_table
    100 | 10377 | 3089 | QUERY   | Undoing 1 transactions on table 133646 with current
 xid 10377 : 10377
    100 | 10378 | 3089 | QUERY   | INSERT INTO test_table values ( $1 )
    100 | 10378 | 3089 | UTILITY | COMMIT
```

Gestion des transactions des procédures stockées en mode non atomique

Une procédure stockée créée en mode NONATOMIC a un comportement de contrôle des transactions différent de celui d'une procédure créée en mode par défaut. À l'instar de la validation automatique des commandes SQL en dehors des procédures stockées, chaque instruction SQL à l'intérieur d'une procédure NONATOMIC s'exécute dans sa propre transaction et est validée automatiquement. Si un utilisateur commence un bloc de transaction explicite dans une procédure stockée NONATOMIC, les instructions SQL contenues dans le bloc ne sont pas automatiquement validées. Le bloc de transaction contrôle la validation ou la restauration des instructions qu'il contient.

Dans les procédures stockées NONATOMIC, vous pouvez ouvrir un bloc de transaction explicite à l'intérieur de la procédure à l'aide de l'instruction START TRANSACTION. Cependant, s'il existe déjà un bloc de transaction ouvert, cette déclaration ne fera rien car Amazon Redshift ne prend pas en charge les sous transactions. La transaction précédente se poursuit.

Lorsque vous utilisez des boucles FOR à l'intérieur d'une procédure NONATOMIC, veillez à ouvrir un bloc de transaction explicite avant de parcourir les résultats d'une requête. Sinon, le curseur est fermé lorsque l'instruction SQL à l'intérieur de la boucle est automatiquement validée.

Voici quelques éléments à prendre en compte lors de l'utilisation du mode de comportement NONATOMIC :

- Chaque instruction SQL contenue dans la procédure stockée est automatiquement validée s'il n'y a pas de bloc de transaction ouvert et si l'auto-validation de la session est définie sur ON.
- Vous pouvez émettre une COMMIT/ROLLBACK/TRUNCATE déclaration pour mettre fin à la transaction si la procédure stockée est appelée depuis un bloc de transactions. Cela n'est pas possible en mode par défaut.
- Vous pouvez lancer une instruction START TRANSACTION pour commencer un bloc de transactions à l'intérieur de la procédure stockée.

Les exemples suivants illustrent le comportement des transactions lors de l'utilisation de procédures stockées NONATOMIC. Pour tous les exemples suivants, la validation automatique est définie sur ON.

Dans l'exemple suivant, une procédure stockée NONATOMIC comporte deux instructions INSERT. Lorsque la procédure est appelée en dehors d'un bloc de transactions, chaque instruction INSERT au sein de la procédure est automatiquement validée.

```
CREATE TABLE test_table_a(v int);
CREATE TABLE test_table_b(v int);

CREATE OR REPLACE PROCEDURE sp_nonatomic_insert_table_a(a int, b int) NONATOMIC AS
$$
BEGIN
    INSERT INTO test_table_a values (a);
    INSERT INTO test_table_b values (b);
END;
$$
LANGUAGE plpgsql;

Call sp_nonatomic_insert_table_a(1,2);

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |               stmt_text
--------+------+------------+---------+----------------------------------------
      1 | 1792 | 1073807554 | UTILITY | Call sp_nonatomic_insert_table_a(1,2);
      1 | 1792 | 1073807554 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1792 | 1073807554 | UTILITY | COMMIT
      1 | 1793 | 1073807554 | QUERY   | INSERT INTO test_table_b values ( $1 )
```

```
      1 | 1793 | 1073807554 | UTILITY | COMMIT
(5 rows)
```

Cependant, lorsque la procédure est appelée à partir d'un bloc BEGIN..COMMIT, toutes les instructions font partie de la même transaction (xid=1799).

```
Begin;
  INSERT INTO test_table_a values (10);
  Call sp_nonatomic_insert_table_a(20,30);
  INSERT INTO test_table_b values (40);
Commit;

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |                stmt_text
--------+------+------------+---------+-----------------------------------------
      1 | 1799 | 1073914035 | UTILITY | Begin;
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_a values (10);
      1 | 1799 | 1073914035 | UTILITY | Call sp_nonatomic_insert_table_a(20,30);
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_b values ( $1 )
      1 | 1799 | 1073914035 | QUERY   | INSERT INTO test_table_b values (40);
      1 | 1799 | 1073914035 | UTILITY | COMMIT
(7 rows)
```

Dans cet exemple, deux instructions INSERT se trouvent entre START TRANSACTION...COMMIT. Lorsque la procédure est appelée en dehors d'un bloc de transaction, les deux instructions INSERT se trouvent dans la même transaction (xid=1866).

```
CREATE OR REPLACE PROCEDURE sp_nonatomic_txn_block(a int, b int) NONATOMIC AS
$$
BEGIN
    START TRANSACTION;
    INSERT INTO test_table_a values (a);
    INSERT INTO test_table_b values (b);
    COMMIT;
END;
$$
LANGUAGE plpgsql;
```

```
Call sp_nonatomic_txn_block(1,2);

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |               stmt_text
--------+------+------------+---------+----------------------------------------
      1 | 1865 | 1073823998 | UTILITY | Call sp_nonatomic_txn_block(1,2);
      1 | 1866 | 1073823998 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1866 | 1073823998 | QUERY   | INSERT INTO test_table_b values ( $1 )
      1 | 1866 | 1073823998 | UTILITY | COMMIT
(4 rows)
```

Lorsque la procédure est appelée à l'intérieur d'un bloc BEGIN...COMMIT, la START TRANSACTION à l'intérieur de la procédure ne fait rien parce qu'il y a déjà une transaction ouverte. La commande COMMIT de la procédure valide la transaction en cours (xid=1876) et en démarre une nouvelle.

```
Begin;
  INSERT INTO test_table_a values (10);
  Call sp_nonatomic_txn_block(20,30);
  INSERT INTO test_table_b values (40);
Commit;

Select userid, xid, pid, type, trim(text) as stmt_text
from svl_statementtext where pid = pg_backend_pid() order by xid , starttime ,
 sequence;

 userid | xid  |    pid     |  type   |               stmt_text
--------+------+------------+---------+----------------------------------------
      1 | 1876 | 1073832133 | UTILITY | Begin;
      1 | 1876 | 1073832133 | QUERY   | INSERT INTO test_table_a values (10);
      1 | 1876 | 1073832133 | UTILITY | Call sp_nonatomic_txn_block(20,30);
      1 | 1876 | 1073832133 | QUERY   | INSERT INTO test_table_a values ( $1 )
      1 | 1876 | 1073832133 | QUERY   | INSERT INTO test_table_b values ( $1 )
      1 | 1876 | 1073832133 | UTILITY | COMMIT
      1 | 1878 | 1073832133 | QUERY   | INSERT INTO test_table_b values (40);
      1 | 1878 | 1073832133 | UTILITY | COMMIT
(8 rows)
```

Cet exemple montre comment travailler avec des boucles de curseur. La table test_table_a possède trois valeurs. L'objectif est de parcourir les trois valeurs et de les insérer dans la table test_table_b.

Si une procédure stockée NONATOMIC est créée de la manière suivante, le curseur « cur1 » n'existe plus après l'exécution de l'instruction INSERT dans la première boucle. En effet, la validation automatique de la commande INSERT ferme le curseur ouvert.

```
insert into test_table_a values (1), (2), (3);

CREATE OR REPLACE PROCEDURE sp_nonatomic_cursor() NONATOMIC
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  cur1 cursor for select * from test_table_a order by 1;
BEGIN
  open cur1;
  Loop
    fetch cur1 into rec;
    exit when not found;
    raise info '%', rec.v;
    insert into test_table_b values (rec.v);
  End Loop;
END
$$;

CALL sp_nonatomic_cursor();

INFO:  1
ERROR:  cursor "cur1" does not exist
CONTEXT:  PL/pgSQL function "sp_nonatomic_cursor" line 7 at fetch
```

Pour que la boucle du curseur fonctionne, placez-la entre START TRANSACTION...COMMIT.

```
insert into test_table_a values (1), (2), (3);

CREATE OR REPLACE PROCEDURE sp_nonatomic_cursor() NONATOMIC
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  cur1 cursor for select * from test_table_a order by 1;
BEGIN
  START TRANSACTION;
  open cur1;
  Loop
```

```
    fetch cur1 into rec;
    exit when not found;
    raise info '%', rec.v;
    insert into test_table_b values (rec.v);
  End Loop;
  COMMIT;
END
$$;

CALL sp_nonatomic_cursor();

INFO:  1
INFO:  2
INFO:  3
CALL
```

Interception des erreurs

Cette rubrique décrit la façon dont Amazon Redshift gère les erreurs.

Lorsqu'une requête ou une commande dans une procédure stockée provoque une erreur, les requêtes suivantes ne s'exécutent pas et la transaction est annulée. Vous pouvez toutefois intercepter les erreurs à l'aide d'un bloc EXCEPTION.

> **Note**
>
> Par défaut, une erreur empêche l'exécution des requêtes suivantes, même si la procédure stockée ne contient pas d'autres conditions génératrices d'erreurs.

```
[ <<label>> ]
[ DECLARE
  declarations ]
BEGIN
  statements
EXCEPTION
  WHEN OTHERS THEN
    statements
END;
```

Lorsqu'une exception se produit et que vous ajoutez un bloc de gestion des exceptions, vous pouvez écrire des instructions RAISE et la plupart des autres instructions PL/PGSQL. Par exemple, vous pouvez générer une exception avec un message personnalisé ou insérer un enregistrement dans une table de journalisation.

Lorsque vous saisissez le bloc de gestion des exceptions, la transaction actuelle est annulée et une nouvelle transaction est créée pour exécuter les instructions dans le bloc. Si les instructions du bloc s'exécutent sans erreur, la transaction est validée et l'exception est renvoyée. Enfin, la procédure stockée se ferme.

La seule condition prise en charge dans un bloc d'exception est OTHERS, qui établit une correspondance avec tout type d'erreur, à l'exception de l'annulation de requête. De plus, si une erreur se produit dans un bloc de gestion des exceptions, elle peut être interceptée par un bloc de gestion des exceptions externe.

Lorsqu'une erreur se produit à l'intérieur de la procédure NONATOMIC, l'erreur n'est pas relancée si elle est gérée par un bloc d'exception. Reportez-vous à la déclaration PL/pgSQL `RAISE` pour lancer une exception capturée par le bloc de gestion des exceptions. Cette déclaration n'est valable que dans les blocs de traitement des exceptions. Pour plus d'informations, consultez RAISE.

Contrôle des conséquences d'une erreur dans une procédure stockée, avec le gestionnaire CONTINUE

Le gestionnaire `CONTINUE` est un type de gestionnaire d'exceptions qui contrôle le flux d'exécution au sein d'une procédure stockée NONATOMIC. En l'utilisant, vous pouvez intercepter et gérer les exceptions sans mettre fin au bloc d'instructions existant. Normalement, lorsqu'une erreur se produit dans une procédure stockée, le flux est interrompu et l'erreur est renvoyée à l'appelant. Cependant, dans certains cas d'utilisation, la condition d'erreur n'est pas suffisamment grave pour justifier l'interruption du flux. Vous souhaiterez peut-être gérer l'erreur de façon fluide, en utilisant la logique de gestion des erreurs de votre choix dans une transaction distincte, puis continuer à exécuter les instructions qui suivent l'erreur. L'exemple suivant montre la syntaxe.

```
[ DECLARE
  declarations ]
BEGIN
  statements
EXCEPTION
  [ CONTINUE_HANDLER | EXIT_HANDLER ] WHEN OTHERS THEN
    handler_statements
END;
```

Plusieurs tables système sont disponibles pour vous aider à recueillir des informations sur les différents types d'erreurs. Pour plus d'informations, consultez STL_LOAD_ERRORS, STL_ERROR et SYS_STREAM_SCAN_ERRORS. Il existe également des tables système supplémentaires que vous pouvez utiliser pour résoudre les erreurs. Pour plus d'informations, consultez Informations de référence sur les tables et les vues système.

exemple

L'exemple suivant montre comment écrire des instructions dans le bloc de gestion des exceptions. La procédure stockée utilise le comportement de gestion des transactions par défaut.

```
CREATE TABLE employee (firstname varchar, lastname varchar);
INSERT INTO employee VALUES ('Tomas','Smith');
CREATE TABLE employee_error_log (message varchar);

CREATE OR REPLACE PROCEDURE update_employee_sp() AS
$$
BEGIN
    UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
    EXECUTE 'select invalid';
EXCEPTION WHEN OTHERS THEN
    RAISE INFO 'An exception occurred.';
    INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
END;
$$
LANGUAGE plpgsql;

CALL update_employee_sp();

INFO:  An exception occurred.
ERROR:  column "invalid" does not exist
CONTEXT:  SQL statement "select invalid"
PL/pgSQL function "update_employee_sp" line 3 at execute statement
```

Dans cet exemple, si nous appelons `update_employee_sp`, le message d'information An exception occurred (Une exception s'est produite) est déclenché et le message d'erreur est généré dans le journal `employee_error_log` de la table de journalisation. L'exception d'origine est renvoyée avant la fin de la procédure stockée. Les requêtes suivantes présentent les enregistrements résultant de l'exécution de l'exemple.

```
SELECT * from employee;
```

```
firstname | lastname
-----------+----------
 Tomas     | Smith

SELECT * from employee_error_log;

          message
-----------------------------------------------
 Error message: column "invalid" does not exist
```

Pour plus d'informations sur RAISE, y compris une aide au formatage et une liste de niveaux supplémentaires, consultez Instructions PL/pgSQL prises en charge.

L'exemple suivant montre comment écrire des instructions dans le bloc de gestion des exceptions. La procédure stockée utilise un comportement de gestion des transactions NONATOMIC. Dans cet exemple, aucune erreur n'est renvoyée à l'appelant à la fin de l'appel de la procédure. L'instruction UPDATE n'est pas restaurée en raison de l'erreur dans l'instruction suivante. Le message d'information est affiché et le message d'erreur est inséré dans la table de journalisation.

```
CREATE TABLE employee (firstname varchar, lastname varchar);
INSERT INTO employee VALUES ('Tomas','Smith');
CREATE TABLE employee_error_log (message varchar);

-- Create the SP in NONATOMIC mode
CREATE OR REPLACE PROCEDURE update_employee_sp_2() NONATOMIC AS
$$
BEGIN
    UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
    EXECUTE 'select invalid';
EXCEPTION WHEN OTHERS THEN
    RAISE INFO 'An exception occurred.';
    INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
END;
$$
LANGUAGE plpgsql;

CALL update_employee_sp_2();
INFO:  An exception occurred.
CALL

SELECT * from employee;
```

```
 firstname | lastname
-----------+----------
 Adam      | Smith
(1 row)

SELECT * from employee_error_log;

                    message
-----------------------------------------------
 Error message: column "invalid" does not exist
(1 row)
```

Cet exemple montre comment créer une procédure avec deux sous-blocs. Lorsque la procédure stockée est appelée, l'erreur du premier sous-bloc est traitée par son bloc de gestion des exceptions. Une fois le premier sous-bloc terminé, la procédure continue d'exécuter le deuxième sous-bloc. Vous pouvez voir dans le résultat qu'aucune erreur n'est déclenchée à la fin de l'appel de la procédure. Les opérations UPDATE et INSERT sur la table employee sont validées. Les messages d'erreur des deux blocs d'exception sont insérés dans la table de journalisation.

```
CREATE TABLE employee (firstname varchar, lastname varchar);
INSERT INTO employee VALUES ('Tomas','Smith');
CREATE TABLE employee_error_log (message varchar);

CREATE OR REPLACE PROCEDURE update_employee_sp_3() NONATOMIC AS
$$
BEGIN
    BEGIN
        UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
        EXECUTE 'select invalid1';
    EXCEPTION WHEN OTHERS THEN
        RAISE INFO 'An exception occurred in the first block.';
        INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
    END;
    BEGIN
        INSERT INTO employee VALUES ('Edie','Robertson');
        EXECUTE 'select invalid2';
    EXCEPTION WHEN OTHERS THEN
        RAISE INFO 'An exception occurred in the second block.';
        INSERT INTO employee_error_log VALUES ('Error message: ' || SQLERRM);
    END;
END;
$$
```

```
LANGUAGE plpgsql;

CALL update_employee_sp_3();
INFO:  An exception occurred in the first block.
INFO:  An exception occurred in the second block.
CALL

SELECT * from employee;

 firstname | lastname
-----------+-----------
 Adam      | Smith
 Edie      | Robertson
(2 rows)

SELECT * from employee_error_log;

                     message
-------------------------------------------------
 Error message: column "invalid1" does not exist
 Error message: column "invalid2" does not exist
(2 rows)
```

L'exemple suivant montre comment utiliser le gestionnaire d'exceptions CONTINUE. Cet exemple crée deux tables et les utilise dans une procédure stockée. Le gestionnaire CONTINUE contrôle le flux d'exécution dans une procédure stockée avec un comportement de gestion des transactions NONATOMIC.

```
CREATE TABLE tbl_1 (a int);
CREATE TABLE tbl_error_logging(info varchar, err_state varchar, err_msg varchar);

CREATE OR REPLACE PROCEDURE sp_exc_handling_1() NONATOMIC AS
$$
BEGIN
    INSERT INTO tbl_1 VALUES (1);
    -- Expect an error for the insert statement following, because of the invalid value
    INSERT INTO tbl_1 VALUES ("val");
    INSERT INTO tbl_1 VALUES (2);
EXCEPTION CONTINUE_HANDLER WHEN OTHERS THEN
    INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
END;
$$ LANGUAGE plpgsql;
```

Appelez la procédure stockée :

```
CALL sp_exc_handling_1();
```

Le flux se déroule comme suit :

1. Une erreur se produit, car une tentative est faite d'insérer un type de données incompatible dans une colonne. Le contrôle passe au bloc EXCEPTION. Lorsque le bloc de gestion des exceptions est entré, la transaction actuelle est annulée et une nouvelle transaction est créée pour exécuter les instructions qu'elle contient.
2. Si les instructions de CONTINUE_HANDLER s'exécutent sans erreur, le contrôle passe à l'instruction qui suit immédiatement celle à l'origine de l'exception. (Si une instruction dans CONTINUE_HANDLER déclenche une nouvelle exception, vous pouvez la gérer avec un gestionnaire d'exceptions dans le bloc EXCEPTION.)

Une fois que vous avez appelé l'exemple de procédure stockée, les tables contiennent les enregistrements suivants :

- Si vous exécutez `SELECT * FROM tbl_1;`, cela renvoie deux enregistrements. Ceux-ci contiennent les valeurs 1 et 2.
- Si vous exécutez `SELECT * FROM tbl_error_logging;`, cela renvoie un enregistrement avec les valeurs suivantes : Encountered error, 42703 et column "val" does not exist in tbl_1.

L'autre exemple suivant de gestion des erreurs utilise à la fois un gestionnaire EXIT et un gestionnaire CONTINUE. Il crée deux tables : une table de données et une table de journalisation. Il crée également une procédure stockée qui illustre la gestion des erreurs :

```
CREATE TABLE tbl_1 (a int);
CREATE TABLE tbl_error_logging(info varchar, err_state varchar, err_msg varchar);

CREATE OR REPLACE PROCEDURE sp_exc_handling_2() NONATOMIC AS
$$
BEGIN
    INSERT INTO tbl_1 VALUES (1);
    BEGIN
        INSERT INTO tbl_1 VALUES (100);
        -- Expect an error for the insert statement following, because of the invalid
 value
```

```
        INSERT INTO tbl_1 VALUES ("val");
        INSERT INTO tbl_1 VALUES (101);
    EXCEPTION EXIT_HANDLER WHEN OTHERS THEN
        INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
    END;
    INSERT INTO tbl_1 VALUES (2);
    -- Expect an error for the insert statement following, because of the invalid value
    INSERT INTO tbl_1 VALUES ("val");
    INSERT INTO tbl_1 VALUES (3);
EXCEPTION CONTINUE_HANDLER WHEN OTHERS THEN
    INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
END;
$$ LANGUAGE plpgsql;
```

Après avoir créé la procédure stockée, appelez-la comme suit :

```
CALL sp_exc_handling_2();
```

Lorsqu'une erreur se produit dans le bloc d'exception interne, qui est placé entre crochets par l'ensemble interne BEGIN et END, elle est gérée par le gestionnaire EXIT. Toutes les erreurs survenant dans le bloc extérieur sont gérées par le gestionnaire CONTINUE.

Une fois que vous avez appelé l'exemple de procédure stockée, les tables contiennent les enregistrements suivants :

- Si vous exécutez `SELECT * FROM tbl_1;`, cela renvoie quatre enregistrements, avec les valeurs 1, 2, 3 et 100.
- Si vous exécutez `SELECT * FROM tbl_error_logging;`, cela renvoie deux enregistrements. Ils ont les valeurs suivantes : Encountered error, 42703 et column "val" does not exist in tbl_1.

Si la table tbl_error_logging n'existe pas, elle déclenche une exception.

L'exemple suivant montre comment utiliser le gestionnaire d'exceptions CONTINUE avec la boucle FOR. Cet exemple crée trois tables et les utilise dans une boucle FOR dans une procédure stockée. La boucle FOR est une variante avec ensemble de résultats, ce qui signifie qu'elle itère sur les résultats d'une requête :

```
CREATE TABLE tbl_1 (a int);
INSERT INTO tbl_1 VALUES (1), (2), (3);
CREATE TABLE tbl_2 (a int);
```

```
CREATE TABLE tbl_error_logging(info varchar, err_state varchar, err_msg varchar);

CREATE OR REPLACE PROCEDURE sp_exc_handling_loop() NONATOMIC AS
$$
DECLARE
 rec RECORD;
BEGIN
    FOR rec IN SELECT a FROM tbl_1
    LOOP
        IF rec.a = 2 THEN
            -- Expect an error for the insert statement following, because of the
 invalid value
            INSERT INTO tbl_2 VALUES("val");
        ELSE
            INSERT INTO tbl_2 VALUES (rec.a);
        END IF;
    END LOOP;
EXCEPTION CONTINUE_HANDLER WHEN OTHERS THEN
    INSERT INTO tbl_error_logging VALUES ('Encountered error', SQLSTATE, SQLERRM);
END;
$$ LANGUAGE plpgsql;
```

Appelez la procédure stockée :

```
CALL sp_exc_handling_loop();
```

Une fois que vous avez appelé l'exemple de procédure stockée, les tables contiennent les enregistrements suivants :

- Si vous exécutez `SELECT * FROM tbl_2;`, cela renvoie deux enregistrements. Ceux-ci contiennent les valeurs 1 et 3.
- Si vous exécutez `SELECT * FROM tbl_error_logging;`, cela renvoie un enregistrement avec les valeurs suivantes : Encountered error, 42703 et column "val" does not exist in tbl_2.

Remarques concernant l'utilisation du gestionnaire CONTINUE :

- Les mots-clés CONTINUE_HANDLER et EXIT_HANDLER ne peuvent être utilisés que dans les procédures stockées NONATOMIC.
- Les mots-clés CONTINUE_HANDLER et EXIT_HANDLER sont facultatifs. EXIT_HANDLER est celui par défaut.

Enregistrement des procédures stockées

Cette rubrique décrit les procédures stockées et les vues qu'Amazon Redshift utilise pour la journalisation des procédures stockées.

Les détails sur les procédures stockées sont enregistrés dans les vues et tables système suivantes :

- SVL_STORED_PROC_CALL : les détails enregistrés concernent l'heure de début et l'heure de fin de l'appel de la procédure stockée, ainsi que l'information selon laquelle l'appel prend fin avant qu'il ne soit terminé. Pour de plus amples informations, veuillez consulter SVL_STORED_PROC_CALL.
- SVL_STORED_PROC_MESSAGES : les messages stockés dans les procédures stockées émises par la requête RAISE sont enregistrés avec le niveau de journalisation correspondant. Pour de plus amples informations, veuillez consulter SVL_STORED_PROC_MESSAGES.
- SVL_QLOG : l'ID de requête de l'appel de procédure est enregistré pour chaque requête appelée à partir d'une procédure stockée. Pour de plus amples informations, veuillez consulter SVL_QLOG.
- STL_UTILITYTEXT : les appels de procédure stockée sont enregistrés une fois qu'ils sont terminés. Pour de plus amples informations, veuillez consulter STL_UTILITYTEXT.
- PG_PROC_INFO : cette vue catalogue système affiche les informations sur les procédures stockées. Pour de plus amples informations, veuillez consulter PG_PROC_INFO.

Limites des procédures stockées

Cette rubrique décrit les limites des procédures stockées Amazon Redshift.

Les considérations suivantes s'appliquent lorsque vous utilisez des procédures stockées Amazon Redshift.

Différences entre Amazon Redshift et PostgreSQL pour la prise en charge des procédures stockées

Voici les différences entre la prise en charge des procédures stockées dans Amazon Redshift et PostgreSQL :

- Amazon Redshift ne prend pas en charge les sous-transactions et, par conséquent, offre une prise en charge limitée pour les blocs de gestion des exceptions.

Considérations et limites

Vous trouverez ci-après des considérations sur les procédures stockées dans Amazon Redshift :

- Le nombre maximal de procédures stockées pour une base de données est de 10 000.
- La taille maximale du code source pour une procédure est de 2 Mo.
- Le nombre maximal de curseurs implicites et explicites que vous pouvez ouvrir simultanément dans une session utilisateur est de 1 (un). Les boucles FOR qui itèrent sur l'ensemble des résultats d'une instruction SQL ouvrent les curseurs implicites. Les curseurs imbriqués ne sont pas pris en charge.
- Les curseurs explicites et implicites ont les mêmes restrictions sur la taille de l'ensemble de résultats que les curseurs Amazon Redshift standard. Pour de plus amples informations, veuillez consulter Contraintes de curseur.
- Le nombre maximal de niveaux pour les appels imbriqués est de 16.
- Le nombre maximal de paramètres de procédure est de 32 pour les arguments en entrée et de 32 pour les arguments en sortie.
- Le nombre maximal de variables dans une procédure stockée est de 1 024.
- Toute commande SQL qui nécessite son propre contexte de transaction n'est pas prise en charge à l'intérieur d'une procédure stockée. En voici quelques exemples :
  - PREPARE
  - CREATE/DROP DATABASE
  - CREATE EXTERNAL TABLE
  - VACUUM
  - SET LOCAL
  - ALTER TABLE APPEND
- L'appel de la méthode `registerOutParameter` via le pilote JDBC (Java Database Connectivity) n'est pas pris en charge pour le type de données `refcursor`. Pour obtenir un exemple de l'utilisation du type de données `refcursor`, consultez Renvoyer un jeu de résultats à partir d'une procédure stockée.

Guide de référence du langage PL/pgSQL

Les procédures stockées dans Amazon Redshift sont basées sur PostgreSQLPL/pgSQL procedural language, with some important differences. In this reference, you can find details of PL/pgSQL

syntax as implemented by Amazon Redshift. For more information about PL/pgSQL, voir PL/pgSQL - langage procédural SQL dans la documentation de PostgreSQL.

Rubriques

- Conventions de référence du langage PL/pgSQL
- Structure de PL/pgSQL
- Instructions PL/pgSQL prises en charge

Conventions de référence du langage PL/pgSQL

Dans cette section, vous trouverez les conventions utilisées pour écrire la syntaxe du langage de procédure stockée PL/pgSQL.

Caractère	Description
CAPS	Les mots en lettres majuscules sont des mots clés.
[ ]	Les crochets indiquent des arguments facultatifs. Plusieurs arguments entre crochets signifient que vous pouvez choisir n'importe quel nombre d'argumen ts. En outre, les arguments entre crochets placés sur des lignes séparées indiquent que l'analyseur Amazon Redshift s'attend à ce que les arguments soient dans l'ordre où ils apparaissent dans la syntaxe.
{ }	Les accolades indiquent que vous devez choisir l'un des arguments proposés.
|	Les barres verticales indiquent que vous pouvez choisir entre les arguments.
red italics	Les mots en italique rouge correspondent à des espaces réservés. Insérez la valeur appropriée à la place du mot en italique rouge.
. . .	Les trois points de suspension indiquent que vous pouvez répéter l'élément précédent.
'	Les mots entre apostrophes droites signifient que vous devez taper les apostrophes.

Structure de PL/pgSQL

PL/pgSQL est un langage procédural avec un grand nombre de constructions communes à d'autres langages procéduraux.

Rubriques

- Bloc
- Déclaration de variable
- Déclaration d'alias
- Variables intégrées
- Types d'enregistrement

Bloc

Instructions PL/pgSQL is a block-structured language. The complete body of a procedure is defined in a block, which contains variable declarations and PL/pgSQL Une instruction peut également être un bloc imbriqué ou un sous-bloc.

Terminez les déclarations et les instructions par un point-virgule. Faites suivre le mot clé END d'un point-virgule dans un bloc ou un sous-bloc. N'utilisez pas de point-virgule après les mots clés DECLARE et BEGIN.

Vous pouvez écrire tous les mots clés et les identifiants en mélangeant majuscules et minuscules. Les identifiants sont implicitement convertis en minuscules à moins d'être placés entre guillemets doubles.

Un tiret double (--) marque le début d'un commentaire qui s'étend jusqu'à la fin de la ligne. Les caractères /* marquent le début d'un commentaire de bloc qui s'étend jusqu'à l'occurrence suivante de */. Vous ne pouvez pas imbriquer des commentaires de bloc. Toutefois, vous pouvez placer des commentaires avec tiret double dans un commentaire de bloc, et un tiret double peut masquer les délimiteurs de commentaire de bloc /* et */.

Toute instruction figurant dans la section des instructions d'un bloc peut être un sous-bloc. Vous pouvez utiliser des sous-blocs pour effectuer un regroupement logique ou pour localiser des variables dans un petit groupe d'instructions.

```
[ <<label>> ]
[ DECLARE
```

```
  declarations ]
BEGIN
  statements
END [ label ];
```

Les variables déclarées dans la section des déclarations précédant un bloc sont initialisées à leur valeur par défaut à chaque entrée dans le bloc. En d'autres termes, elles ne sont pas initialisées une seule fois par appel de fonction.

Vous en trouverez un exemple ci-dessous.

```
CREATE PROCEDURE update_value() AS $$
DECLARE
  value integer := 20;
BEGIN
  RAISE NOTICE 'Value here is %', value;  -- Value here is 20
  value := 50;
  --
  -- Create a subblock
  --
  DECLARE
    value integer := 80;
  BEGIN
    RAISE NOTICE 'Value here is %', value;  -- Value here is 80
  END;

  RAISE NOTICE 'Value here is %', value;  -- Value here is 50
END;
$$ LANGUAGE plpgsql;
```

Utilisez une étiquette pour identifier le bloc à utiliser dans une instruction EXIT ou pour qualifier les noms des variables déclarées dans le bloc.

Ne confondez pas l'utilisation de BEGIN/END for grouping statements in PL/pgSQL with the database commands for transaction control. The BEGIN and END in PL/pgSQL are uniquement pour le regroupement. Ils ne démarrent pas et ne finissent pas une transaction.

Déclaration de variable

Déclarez toutes les variables d'un bloc, à l'exception des variables de boucle, dans la section DECLARE du bloc. Les variables peuvent utiliser un type de données Amazon Redshift quelconque valide. Pour connaître les types de données pris en charge, consultez Types de données.

Les variables PL/pgSQL peuvent être d'un type de données quelconque pris en charge par Amazon Redshift, plus `RECORD` et `refcursor`. Pour plus d'informations sur `RECORD`, consultez Types d'enregistrement. Pour plus d'informations sur `refcursor`, consultez Curseurs.

```
DECLARE
name [ CONSTANT ] type [ NOT NULL ] [ { DEFAULT | := } expression ];
```

Vous trouverez ci-après des exemples de déclarations de variables.

```
customerID integer;
numberofitems numeric(6);
link varchar;
onerow RECORD;
```

La variable de boucle d'une boucle FOR effectuant une itération sur une plage d'entiers est automatiquement déclarée en tant que variable de type entier.

La clause DEFAULT, si elle est spécifiée, fournit la valeur initiale attribuée à la variable lors de l'entrée dans le bloc. Si la clause DEFAULT n'est pas spécifiée, la variable est initialisée avec la valeur SQL NULL. L'option CONSTANT empêche l'attribution d'une valeur à la variable, afin que sa valeur reste constante pendant toute la durée du bloc. Si NOT NULL est spécifié, l'attribution d'une valeur null entraîne une erreur d'exécution. Toutes les variables déclarées comme NOT NULL doivent avoir une valeur par défaut non null spécifiée.

La valeur par défaut est évaluée à chaque entrée dans le bloc. Par exemple, l'attribution de `now()` à une variable de type `timestamp` fait que la variable contiendra l'heure de l'appel actuel de la fonction, et non pas l'heure de la précompilation de la fonction.

```
quantity INTEGER DEFAULT 32;
url VARCHAR := 'http://mysite.com';
user_id CONSTANT INTEGER := 10;
```

Le type de données `refcursor` est le type de données des variables de curseur au sein des procédures stockées. Une valeur `refcursor` peut être retournée d'une procédure stockée. Pour de plus amples informations, veuillez consulter Renvoyer un jeu de résultats à partir d'une procédure stockée.

Déclaration d'alias

Si la signature d'une procédure stockée omet le nom de l'argument, vous pouvez déclarer un alias pour cet argument.

```
name ALIAS FOR $n;
```

Variables intégrées

Les variables intégrées suivantes sont prises en charge :

- FOUND
- SQLSTATE
- SQLERRM
- GET DIAGNOSTICS integer_var := ROW_COUNT;

FOUND est une variable spéciale de type booléen. FOUND commence avec la valeur false au sein de chaque appel de procédure. La variable FOUND est définie par les types d'instructions suivants :

- SELECT INTO

  Attribue à FOUND la valeur true si une ligne est renvoyée, et la valeur false si aucune ligne n'est renvoyée.
- UPDATE, INSERT et DELETE

  Attribue à FOUND la valeur true si au moins une ligne est affectée, et la valeur false si aucune ligne n'est affectée.
- FETCH

  Attribue à FOUND la valeur true si une ligne est renvoyée, et la valeur false si aucune ligne n'est renvoyée.
- Instruction FOR

  Attribue à FOUND la valeur true si l'instruction FOR effectue une ou plusieurs itérations, sinon la valeur false. Cela s'applique aux trois variantes de l'instruction FOR : les boucles FOR de type entier, les boucles FOR de type jeu d'enregistrements et les boucles FOR de type jeu d'enregistrements dynamique.

FOUND est définie à la sortie de la boucle FOR. Pendant l'exécution de la boucle, la variable FOUND n'est pas modifiée par l'instruction FOR. Toutefois, elle peut être modifiée en exécutant d'autres instructions dans le corps de la boucle.

Vous en trouverez un exemple ci-dessous.

```
CREATE TABLE employee(empname varchar);
CREATE OR REPLACE PROCEDURE show_found()
AS  $$
DECLARE
  myrec record;
BEGIN
  SELECT INTO myrec * FROM employee WHERE empname = 'John';
  IF NOT FOUND THEN
    RAISE EXCEPTION 'employee John not found';
  END IF;
END;
$$ LANGUAGE plpgsql;
```

Dans un gestionnaire d'exceptions, la variable spéciale SQLSTATE contient le code d'erreur correspondant à l'exception qui a été levée. La variable spéciale SQLERRM contient le message d'erreur associé à l'exception. Ces variables sont indéfinies en dehors des gestionnaires d'exception et affichent une erreur si elles sont utilisées.

Vous en trouverez un exemple ci-dessous.

```
CREATE OR REPLACE PROCEDURE sqlstate_sqlerrm() AS
$$
BEGIN
  UPDATE employee SET firstname = 'Adam' WHERE lastname = 'Smith';
  EXECUTE 'select invalid';
  EXCEPTION WHEN OTHERS THEN
  RAISE INFO 'error message SQLERRM %', SQLERRM;
  RAISE INFO 'error message SQLSTATE %', SQLSTATE;
END;
$$ LANGUAGE plpgsql;
```

ROW_COUNT est utilisée avec la commande GET DIAGNOSTICS. Elle indique le nombre de lignes traitées par la dernière commande SQL envoyée au moteur SQL.

Vous en trouverez un exemple ci-dessous.

```
CREATE OR REPLACE PROCEDURE sp_row_count() AS
$$
DECLARE
  integer_var int;
BEGIN
  INSERT INTO tbl_row_count VALUES(1);
  GET DIAGNOSTICS integer_var := ROW_COUNT;
  RAISE INFO 'rows inserted = %', integer_var;
END;
$$ LANGUAGE plpgsql;
```

Types d'enregistrement

Un type RECORD n'est pas un vrai type de données, seulement un espace réservé. Les variables de type d'enregistrement assument la structure de ligne réelle de la ligne à laquelle elles sont attribuées au cours de l'exécution d'une commande SELECT ou FOR. La sous-structure d'une variable d'enregistrement peut changer chaque fois qu'une valeur lui est attribuée. Tant qu'une variable d'enregistrement n'a pas été attribuée, elle n'a aucune sous-structure. Toute tentative d'accès à un champ compris dedans génère une erreur d'exécution.

```
name RECORD;
```

Vous en trouverez un exemple ci-dessous.

```
CREATE TABLE tbl_record(a int, b int);
INSERT INTO tbl_record VALUES(1, 2);
CREATE OR REPLACE PROCEDURE record_example()
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
BEGIN
  FOR rec IN SELECT a FROM tbl_record
  LOOP
    RAISE INFO 'a = %', rec.a;
  END LOOP;
END;
$$;
```

Instructions PL/pgSQL prises en charge

Les instructions PL/pgSQL complémentent les commandes SQL avec des constructions procédurales, y compris des expressions de boucle et conditionnelles, pour contrôler le flux logique. La plupart des commandes SQL peuvent être utilisées, y compris celles du langage de manipulation de données (DML) telles que COPY, UNLOAD et INSERT, et celles du langage de définition de données (DDL) telles que CREATE TABLE. Pour obtenir la liste complète des commandes SQL, veuillez consulter Commandes SQL. De plus, les instructions PL/pgSQL suivantes sont prises en charge par Amazon Redshift.

Rubriques

- Affectation
- SELECT INTO
- No-op
- Instructions SQL dynamiques
- Return
- Conditions : IF
- Conditions : CASE
- Boucles
- Curseurs
- RAISE
- Contrôle de transaction

Affectation

L'instruction d'affectation affecte une valeur à une variable. L'expression doit renvoyer une valeur unique.

```
identifier := expression;
```

L'utilisation de l'élément non standard = pour l'affectation, à la place de :=, est également acceptée.

Si le type de données de l'expression ne correspond pas au type de données de la variable ou si la variable a une taille ou une précision, la valeur obtenue est implicitement convertie.

Des exemples sont fournis ci-dessous.

```
customer_number := 20;
tip := subtotal * 0.15;
```

SELECT INTO

L'instruction SELECT INTO affecte le résultat de plusieurs colonnes (mais d'une seule ligne) dans une variable d'enregistrement ou une liste de variables scalaires.

```
SELECT INTO target select_expressions FROM ...;
```

Dans la syntaxe précédente, il *target* peut s'agir d'une variable d'enregistrement ou d'une liste séparée par des virgules de variables simples et de champs d'enregistrement. La *select_expressions* liste et le reste de la commande sont identiques à ceux du SQL normal.

Si une liste de variables est utilisée en tant que*target*, les valeurs sélectionnées doivent correspondre exactement à la structure de la cible, sinon une erreur d'exécution se produit. Lorsqu'une variable d'enregistrement est la cible, elle se configure automatiquement sur le type de ligne des colonnes de résultat de la requête.

La clause INTO peut apparaître presque partout dans l'instruction SELECT. Elle apparaît habituellement juste après la clause SELECT ou juste avant la clause FROM. C'est-à-dire qu'il apparaît juste avant ou juste après la *select_expressions* liste.

Si la requête ne renvoie aucune ligne, des valeurs NULL sont attribuées à*target*. Si la requête renvoie plusieurs lignes, la première ligne est affectée à *target* et les autres sont ignorées. À moins que l'instruction contienne une commande ORDER BY, la première ligne n'est pas déterministe.

Pour déterminer si l'affectation a retourné au moins une ligne, utilisez la variable FOUND spéciale.

```
SELECT INTO customer_rec * FROM cust WHERE custname = lname;
IF NOT FOUND THEN
  RAISE EXCEPTION 'employee % not found', lname;
END IF;
```

Pour tester si un résultat d'enregistrement est null, vous pouvez utiliser l'opérateur conditionnel IS NULL. Aucune méthode ne permet de déterminer si des lignes supplémentaires peuvent avoir été ignorées. L'exemple suivant traite le cas où aucune ligne n'a été renvoyée.

```
CREATE OR REPLACE PROCEDURE select_into_null(return_webpage OUT varchar(256))
AS $$
```

```
DECLARE
  customer_rec RECORD;
BEGIN
  SELECT INTO customer_rec * FROM users WHERE user_id=3;
  IF customer_rec.webpage IS NULL THEN
    -- user entered no webpage, return "http://"
    return_webpage = 'http://';
  END IF;
END;
$$ LANGUAGE plpgsql;
```

No-op

L'instruction no-op (`NULL;`) est une instruction d'espace réservé qui ne fait rien. Une instruction no-op peut indiquer qu'une branche d'une IF-THEN-ELSE chaîne est vide.

```
NULL;
```

Instructions SQL dynamiques

Pour générer des commandes dynamiques qui peuvent impliquer différentes tables ou différents types de données chaque fois qu'elles sont exécutées à partir d'une procédure stockée PL/pgSQL, utilisez l'instruction `EXECUTE`.

```
EXECUTE command-string [ INTO target ];
```

Dans ce qui précède, *`command-string`* une expression produit une chaîne (de type texte) contenant la commande à exécuter. Cette *`command-string`* valeur est envoyée au moteur SQL. Aucune substitution des variables PL/pgSQL n'est effectuée sur la chaîne de commande. Les valeurs des variables doivent être insérées dans la chaîne de commande lorsqu'elle est construite.

> **Note**
>
> Vous ne pouvez pas utiliser les instructions COMMIT et ROLLBACK depuis SQL dynamique. Pour plus d'informations sur l'utilisation des instructions COMMIT et ROLLBACK dans une procédure stockée, consultez Gestion des transactions.

Lorsque vous utilisez des commandes dynamiques, vous devez souvent traiter l'échappement des guillemets simples. Nous vous recommandons de placer le texte fixe entre guillemets dans le corps

de votre fonction en utilisant des guillemets dollar. Les valeurs dynamiques à insérer dans une requête construite nécessitent un traitement spécial, car elles peuvent contenir elles-mêmes des guillemets. L'exemple suivant suppose des guillemets dollar pour la fonction dans son ensemble, afin que les guillemets n'aient pas besoin d'être doublés.

```
EXECUTE 'UPDATE tbl SET '
  || quote_ident(colname)
  || ' = '
  || quote_literal(newvalue)
  || ' WHERE key = '
  || quote_literal(keyvalue);
```

L'exemple précédent illustre les fonctions `quote_ident(text)` et `quote_literal(text)`. Cet exemple transmet des variables contenant des identifiants de colonne et de table à la fonction `quote_ident`. Il transmet également des variables qui contiennent des chaînes littérales dans la commande construite à la fonction `quote_literal`. Ces deux fonctions prennent les mesures appropriées pour retourner le texte d'entrée entre guillemets doubles ou simples respectivement, avec n'importe quels caractères spéciaux intégrés correctement échappés.

Les guillemets dollar sont utiles uniquement pour citer du texte fixe. N'écrivez pas l'exemple précédent dans le format suivant.

```
EXECUTE 'UPDATE tbl SET '
  || quote_ident(colname)
  || ' = $$'
  || newvalue
  || '$$ WHERE key = '
  || quote_literal(keyvalue);
```

En effet, cet exemple s'interrompt si le contenu de `newvalue` contient $$. Le même problème s'applique à tout autre délimiteur de guillemets dollar que vous pouvez choisir. Pour citer en toute sécurité du texte qui n'est pas connu à l'avance, utilisez la fonction `quote_literal`.

Return

L'instruction RETURN retourne à l'appelant à partir d'une procédure stockée.

```
RETURN;
```

Vous en trouverez un exemple ci-dessous.

```
CREATE OR REPLACE PROCEDURE return_example(a int)
AS $$
BEGIN
  FOR b in 1..10 LOOP
    IF b < a THEN
      RAISE INFO 'b = %', b;
    ELSE
      RETURN;
    END IF;
  END LOOP;
END;
$$ LANGUAGE plpgsql;
```

Conditions : IF

L'instruction conditionnelle IF peut prendre les formes suivantes dans le langage PL/pgSQL qu'Amazon Redshift utilise :

- IF ... THEN

  ```
  IF boolean-expression THEN
    statements
  END IF;
  ```

  Vous en trouverez un exemple ci-dessous.

  ```
  IF v_user_id <> 0 THEN
    UPDATE users SET email = v_email WHERE user_id = v_user_id;
  END IF;
  ```

- IF ... THEN ... ELSE

  ```
  IF boolean-expression THEN
    statements
  ELSE
    statements
  END IF;
  ```

  Vous en trouverez un exemple ci-dessous.

  ```
  IF parentid IS NULL OR parentid = ''
  ```

```
THEN
  return_name = fullname;
  RETURN;
ELSE
  return_name = hp_true_filename(parentid) || '/' || fullname;
  RETURN;
END IF;
```

- IF ... THEN ... ELSIF ... THEN ... ELSE

  Le mot clé ELSIF peut également être orthographié ELSEIF.

```
IF boolean-expression THEN
  statements
[ ELSIF boolean-expression THEN
  statements
[ ELSIF boolean-expression THEN
  statements
    ...] ]
[ ELSE
  statements ]
END IF;
```

  Vous en trouverez un exemple ci-dessous.

```
IF number = 0 THEN
  result := 'zero';
ELSIF number > 0 THEN
  result := 'positive';
ELSIF number < 0 THEN
  result := 'negative';
ELSE
  -- the only other possibility is that number is null
  result := 'NULL';
END IF;
```

Conditions : CASE

L'instruction conditionnelle CASE peut prendre les formes suivantes dans le langage PL/pgSQL qu'Amazon Redshift utilise :

- CASE simple

```
CASE search-expression
WHEN expression [, expression [ ... ]] THEN
  statements
[ WHEN expression [, expression [ ... ]] THEN
  statements
  ... ]
[ ELSE
  statements ]
END CASE;
```

Une instruction CASE simple fournit une exécution conditionnelle basée sur l'égalité des opérandes.

La *search-expression* valeur est évaluée une fois et comparée successivement à chacune des clauses *expression* WHEN. Si une correspondance est trouvée, l'*statements*exécution correspondante est exécutée, puis le contrôle passe à l'instruction suivante après END CASE. Les expressions WHEN suivantes ne sont pas évaluées. Si aucune correspondance n'est trouvée, le ELSE *statements* s'exécute. Toutefois, en l'absence de ELSE, une exception CASE_NOT_FOUND est levée.

Vous en trouverez un exemple ci-dessous.

```
CASE x
WHEN 1, 2 THEN
  msg := 'one or two';
ELSE
  msg := 'other value than one or two';
END CASE;
```

- CASE avec recherche

```
CASE
WHEN boolean-expression THEN
  statements
[ WHEN boolean-expression THEN
  statements
  ... ]
[ ELSE
  statements ]
```

```
END CASE;
```

La forme de CASE avec recherche fournit une exécution conditionnelle basée sur la véracité des expressions booléennes.

Chaque clause WHEN *boolean-expression* est évaluée à tour de rôle, jusqu'à ce que l'on trouve une clause vraie. Les instructions correspondantes s'exécutent alors, puis le contrôle passe à l'instruction qui suit END CASE. Les QUAND suivants *expressions* ne sont pas évalués. Si aucun résultat réel n'est trouvé, les ELSE *statements* sont exécutés. Toutefois, en l'absence de ELSE, une exception CASE_NOT_FOUND est levée.

Vous en trouverez un exemple ci-dessous.

```
CASE
WHEN x BETWEEN 0 AND 10 THEN
  msg := 'value is between zero and ten';
WHEN x BETWEEN 11 AND 20 THEN
  msg := 'value is between eleven and twenty';
END CASE;
```

Boucles

Les instructions de boucle peuvent prendre les formes suivantes dans le langage PL/pgSQL qu'Amazon Redshift utilise :

- Boucle simple

```
[<<label>>]
LOOP
  statements
END LOOP [ label ];
```

Une boucle simple définit une boucle sans condition qui se répète indéfiniment jusqu'à ce qu'elle soit terminée par une instruction EXIT ou RETURN. Une étiquette facultative peut être utilisée par les instructions EXIT et CONTINUE au sein de boucles imbriquées pour spécifier à quelle boucle l'instruction EXIT ou CONTINUE fait référence.

Vous en trouverez un exemple ci-dessous.

```
CREATE OR REPLACE PROCEDURE simple_loop()
LANGUAGE plpgsql
AS $$
BEGIN
  <<simple_while>>
  LOOP
    RAISE INFO 'I am raised once';
    EXIT simple_while;
    RAISE INFO 'I am not raised';
  END LOOP;
  RAISE INFO 'I am raised once as well';
END;
$$;
```

- Quitter une boucle

```
EXIT [ label ] [ WHEN expression ];
```

Si *label* ce n'est pas le cas, la boucle la plus interne est interrompue et l'instruction qui suit le END LOOP s'exécute ensuite. S'il *label* est présent, il doit s'agir de l'étiquette du niveau actuel ou d'un niveau externe de boucle ou de bloc imbriqué. La boucle ou le bloc nommé est alors interrompu et le contrôle passe à l'instruction située après le END correspondant à la boucle ou au bloc.

Si WHEN est spécifié, la sortie de boucle ne se produit que si elle *expression* est vraie. Dans le cas contraire, le contrôle passe à l'instruction qui suit EXIT.

Vous pouvez utiliser EXIT avec tous les types de boucle. Il n'est pas limité à une utilisation avec des boucles sans condition.

Lorsqu'il est utilisé avec un bloc BEGIN, EXIT passe le contrôle à l'instruction suivante située après la fin du bloc. Une étiquette doit être utilisée à cet effet. Une instruction EXIT sans étiquette n'est jamais considérée comme correspondant à un bloc BEGIN.

Vous en trouverez un exemple ci-dessous.

```
CREATE OR REPLACE PROCEDURE simple_loop_when(x int)
LANGUAGE plpgsql
AS $$
DECLARE i INTEGER := 0;
```

```
BEGIN
  <<simple_loop_when>>
  LOOP
    RAISE INFO 'i %', i;
    i := i + 1;
    EXIT simple_loop_when WHEN (i >= x);
  END LOOP;
END;
$$;
```

- Continuer une boucle

```
CONTINUE [ label ] [ WHEN expression ];
```

Si *label* ce n'est pas le cas, l'exécution passe à l'itération suivante de la boucle la plus interne. Ainsi, toutes les instructions restantes du corps de la boucle sont ignorées. Le contrôle retourne ensuite à l'expression de contrôle de boucle (le cas échéant) pour déterminer si une autre itération de boucle est requise. S'il *label* est présent, il indique le label de la boucle dont l'exécution est poursuivie.

Si WHEN est spécifié, l'itération suivante de la boucle ne commence que si elle *expression* est vraie. Dans le cas contraire, le contrôle passe à l'instruction qui suit CONTINUE.

Vous pouvez utiliser CONTINUE avec tous les types de boucle. Il n'est pas limité à une utilisation avec des boucles sans condition.

```
CONTINUE mylabel;
```

- Boucle WHILE

```
[<<label>>]
WHILE expression LOOP
  statements
END LOOP [ label ];
```

L'instruction WHILE répète une séquence d'instructions tant qu'elles sont *boolean-expression* évaluées comme vraies. L'expression est vérifiée juste avant chaque entrée dans le corps de la boucle.

Vous en trouverez un exemple ci-dessous.

```
WHILE amount_owed > 0 AND gift_certificate_balance > 0 LOOP
  -- some computations here
END LOOP;

WHILE NOT done LOOP
  -- some computations here
END LOOP;
```

- Boucle FOR (variante avec entier)

```
[<<label>>]
FOR name IN [ REVERSE ] expression .. expression LOOP
  statements
END LOOP [ label ];
```

La boucle FOR (variante avec entier) crée une boucle qui effectue des itérations sur une plage de valeurs entières. Le nom de la variable est défini automatiquement de type entier et existe uniquement au sein de la boucle. Toute définition existante du nom de la variable est ignorée au sein de la boucle. Les deux expressions qui donnent les limites inférieure et supérieure de la plage sont évaluées une fois à l'entrée de la boucle. Si vous spécifiez REVERSE, la valeur du pas est soustraite au lieu d'être ajoutée après chaque itération.

Si la limite inférieure est supérieure à la limite supérieure (ou inférieure à celle-ci si REVERSE est utilisé), le corps de la boucle ne s'exécute pas. Aucune erreur n'est levée.

Si une étiquette est attachée à la boucle FOR, Vous pouvez référencer la variable entière de boucle avec un nom complet en utilisant cette étiquette.

Vous en trouverez un exemple ci-dessous.

```
FOR i IN 1..10 LOOP
  -- i will take on the values 1,2,3,4,5,6,7,8,9,10 within the loop
END LOOP;

FOR i IN REVERSE 10..1 LOOP
  -- i will take on the values 10,9,8,7,6,5,4,3,2,1 within the loop
END LOOP;
```

- Boucle FOR (variante avec ensemble de résultats)

```
[<<label>>]
FOR target IN query LOOP
  statements
END LOOP [ label ];
```

*target*Il s'agit d'une variable d'enregistrement ou d'une liste de variables scalaires séparées par des virgules. La cible se voit successivement attribuer chaque ligne résultant de la requête et le corps de la boucle est exécuté pour chaque ligne.

La boucle FOR (variante avec ensemble de résultats) permet à une procédure stockée d'itérer sur les résultats d'une requête et de manipuler ces données en conséquence.

Vous en trouverez un exemple ci-dessous.

```
CREATE PROCEDURE cs_refresh_reports() AS $$
DECLARE
  reports RECORD;
BEGIN
  FOR reports IN SELECT * FROM cs_reports ORDER BY sort_key LOOP
    -- Now "reports" has one record from cs_reports
    EXECUTE 'INSERT INTO ' || quote_ident(reports.report_name) || ' ' ||
 reports.report_query;
  END LOOP;
  RETURN;
END;
$$ LANGUAGE plpgsql;
```

- Boucle FOR avec instruction SQL dynamique

```
[<<label>>]
FOR record_or_row IN EXECUTE text_expression LOOP
  statements
END LOOP;
```

Une boucle FOR avec instruction SQL dynamique permet à une procédure stockée d'itérer sur les résultats d'une requête dynamique et de manipuler ces données en conséquence.

Vous en trouverez un exemple ci-dessous.

```
CREATE OR REPLACE PROCEDURE for_loop_dynamic_sql(x int)
```

```
LANGUAGE plpgsql
AS $$
DECLARE
  rec RECORD;
  query text;
BEGIN
  query := 'SELECT * FROM tbl_dynamic_sql LIMIT ' || x;
  FOR rec IN EXECUTE query
  LOOP
    RAISE INFO 'a %', rec.a;
  END LOOP;
END;
$$;
```

Curseurs

Plutôt que d'exécuter immédiatement une requête entière, vous pouvez configurer un curseur. Un curseur encapsule une requête et lit le résultat de la requête en traitant plusieurs lignes à la fois. Une des raisons de procéder ainsi est d'éviter un dépassement de mémoire lorsque le résultat contient un grand nombre de lignes. Une autre raison est de retourner une référence à un curseur qu'une procédure stockée a créé, ce qui permet à l'appelant de lire les lignes. Cette approche fournit une méthode efficace pour retourner de grands ensembles de lignes à partir de procédures stockées.

Pour utiliser les curseurs dans une procédure stockée NONATOMIC, placez la boucle du curseur entre START TRANSACTION...COMMIT.

Pour configurer un curseur, commencez par déclarer une variable de curseur. Tout accès aux curseurs dans PL/pgSQL passe par les variables de curseur, qui sont toujours du type de données spécial `refcursor`. Un type de données `refcursor` détient simplement une référence à un curseur.

Vous pouvez créer une variable de curseur en la déclarant en tant que variable de type `refcursor`. Vous pouvez également utiliser la syntaxe de déclaration de curseur suivante.

```
name CURSOR [ ( arguments ) ] FOR query ;
```

Dans ce qui précède, *`arguments`* (si spécifié) se trouve une liste de *`name datatype`* paires séparées par des virgules qui définissent chacune des noms à remplacer par des valeurs de paramètres. *`query`* Les valeurs réelles à substituer à ces noms sont spécifiées ultérieurement, à l'ouverture du curseur.

Des exemples sont fournis ci-dessous.

```
DECLARE
  curs1 refcursor;
  curs2 CURSOR FOR SELECT * FROM tenk1;
  curs3 CURSOR (key integer) IS SELECT * FROM tenk1 WHERE unique1 = key;
```

Ces trois variables ont le type de données `refcursor`, mais la première peut être utilisée avec une requête quelconque. Au contraire, une requête entièrement spécifiée est liée à la seconde, et une requête paramétrée est liée à la dernière. La valeur `key` est remplacée par une valeur de paramètre entière à l'ouverture du curseur. La variable `curs1` est dite non liée, car elle n'est liée à aucune requête particulière.

Avant de pouvoir utiliser un curseur pour récupérer des lignes, vous devez l'ouvrir. Le langage PL/pgSQL a trois formes d'instruction OPEN, dont deux utilisent des variables de curseur non liées, tandis que la troisième utilise une variable de curseur liée :

- Ouverture pour sélection : la variable de curseur est ouverte et reçoit la requête spécifiée à exécuter. Le curseur ne doit pas être déjà ouvert. De plus, il doit avoir été déclaré en tant que curseur non lié (c'est-à-dire, en tant que simple variable `refcursor`). La requête SELECT est traitée de la même manière que les autres instructions SELECT dans le langage PL/pgSQL.

  ```
  OPEN cursor_name FOR SELECT ...;
  ```

  Vous en trouverez un exemple ci-dessous.

  ```
  OPEN curs1 FOR SELECT * FROM foo WHERE key = mykey;
  ```

- Ouverture pour exécution : la variable de curseur est ouverte et reçoit la requête spécifiée à exécuter. Le curseur ne doit pas être déjà ouvert. De plus, il doit avoir été déclaré en tant que curseur non lié (c'est-à-dire, en tant que simple variable `refcursor`). La requête est spécifiée en tant qu'expression de chaîne de la même manière que dans la commande EXECUTE. Cette approche est flexible et permet à la requête de varier d'une exécution à l'autre.

  ```
  OPEN cursor_name FOR EXECUTE query_string;
  ```

  Vous en trouverez un exemple ci-dessous.

```
OPEN curs1 FOR EXECUTE 'SELECT * FROM ' || quote_ident($1);
```

- Ouverture d'un curseur lié : cette forme d'instruction OPEN est utilisée pour ouvrir une variable de curseur à laquelle sa requête a été liée quand elle a été déclarée. Le curseur ne doit pas être déjà ouvert. La liste des expressions des valeurs d'argument réelles doit apparaître si et seulement si le curseur a été déclaré comme acceptant des arguments. Ces valeurs sont substituées dans la requête.

  ```
  OPEN bound_cursor_name [ ( argument_values ) ];
  ```

  Vous en trouverez un exemple ci-dessous.

  ```
  OPEN curs2;
  OPEN curs3(42);
  ```

Une fois qu'un curseur a été ouvert, vous pouvez l'utiliser à l'aide des instructions décrites ci-après. Ces instructions ne sont pas tenues de figurer dans la procédure stockée qui a ouvert le curseur. Vous pouvez retourner une valeur `refcursor` à partir d'une procédure stockée et laisser l'appelant opérer sur le curseur. Tous les portails sont implicitement fermés à la fin de la transaction. Par conséquent, vous pouvez utiliser une valeur `refcursor` pour référencer un curseur ouvert seulement jusqu'à la fin de la transaction.

- L'instruction FETCH extrait la ligne suivante du curseur dans une cible. Cette cible peut être une variable de ligne, une variable d'enregistrement ou une liste séparée par des virgules de variables simples, comme avec SELECT INTO. Comme avec SELECT INTO, vous pouvez vérifier la variable spéciale FOUND pour voir si une ligne a été obtenue.

  ```
  FETCH cursor INTO target;
  ```

  Vous en trouverez un exemple ci-dessous.

  ```
  FETCH curs1 INTO rowvar;
  ```

- L'instruction CLOSE ferme le portail sous-jacent à un curseur ouvert. Vous pouvez utiliser cette instruction pour libérer des ressources avant la fin de la transaction. Vous pouvez également

utiliser cette instruction pour libérer la variable de curseur, qui pourra ainsi être de nouveau ouverte.

```
CLOSE cursor;
```

Vous en trouverez un exemple ci-dessous.

```
CLOSE curs1;
```

RAISE

Utilisez l'instruction `RAISE level` pour signaler les messages et les erreurs.

```
RAISE level 'format' [, variable [, ...]];
```

Les niveaux possibles sont NOTICE, INFO, LOG, WARNING et EXCEPTION. EXCEPTION lève une erreur, ce qui entraîne normalement l'annulation de la transaction en cours. Les autres niveaux génèrent seulement des messages de différents niveaux de priorité.

Dans la chaîne de format, % est remplacé par la représentation de chaîne de l'argument facultatif suivant. Écrivez %% pour obtenir un % littéral. Actuellement, les arguments facultatifs doivent être des variables simples et non pas des expressions, et le format doit être un littéral de chaîne simple.

Dans l'exemple suivant, la valeur de `v_job_id` remplace le % dans la chaîne.

```
RAISE NOTICE 'Calling cs_create_job(%)', v_job_id;
```

Utilisez l'instruction `RAISE` pour relancer l'exception capturée par un bloc de gestion des exceptions. Cette déclaration n'est valable que dans les blocs de gestion des exceptions des procédures stockées en mode NONATOMIC.

```
RAISE;
```

Contrôle de transaction

Vous pouvez utiliser les instructions de contrôle de transaction du langage PL/pgSQL qu'Amazon Redshift utilise. Pour plus d'informations sur l'utilisation des instructions COMMIT, ROLLBACK et TRUNCATE dans une procédure stockée, consultez Gestion des transactions.

Dans les procédures stockées en mode NONATOMIC, utilisez `START TRANSACTION` pour démarrer un bloc de transaction.

```
START TRANSACTION;
```

> **Note**
>
> L'instruction PL/pgSQL START TRANSACTION est différente de la commande SQL START TRANSACTION pour les raisons suivantes :
>
> - Dans les procédures stockées, START TRANSACTION n'est pas synonyme de BEGIN.
> - L'instruction PL/pgSQL ne prend pas en charge les mots-clés facultatifs de niveau d'isolement et d'autorisations d'accès.

Vues matérialisées dans Amazon Redshift

Cette section explique comment créer et utiliser des vues matérialisées dans Amazon Redshift. Une vue matérialisée est un objet de base de données qui stocke les résultats d'une requête, qui peuvent être utilisés pour améliorer les performances et l'efficacité.

Dans un environnement d’entrepôt des données, les applications doivent souvent effectuer des requêtes complexes sur de grandes tables. Il y a notamment les instructions SELECT, qui effectuent des jointures et des agrégations de plusieurs tables sur des tables contenant des milliards de lignes. Le traitement de ces requêtes peut être coûteux en termes de ressources système et de temps nécessaires pour calculer les résultats.

Les vues matérialisées dans Amazon Redshift permettent de résoudre ces problèmes. Une vue matérialisée contient un ensemble de résultats précalculés basés sur une requête SQL sur une ou plusieurs tables de base. Vous pouvez émettre des instructions SELECT pour interroger une vue matérialisée, de la même manière que vous pouvez interroger d’autres tables ou vues de la base de données. Amazon Redshift renvoie les résultats précalculés à partir de la vue matérialisée sans forcément accéder aux tables de base. Du point de vue de l’utilisateur, les résultats de la requête sont renvoyés bien plus rapidement par rapport à l’extraction des mêmes données des tables de base.

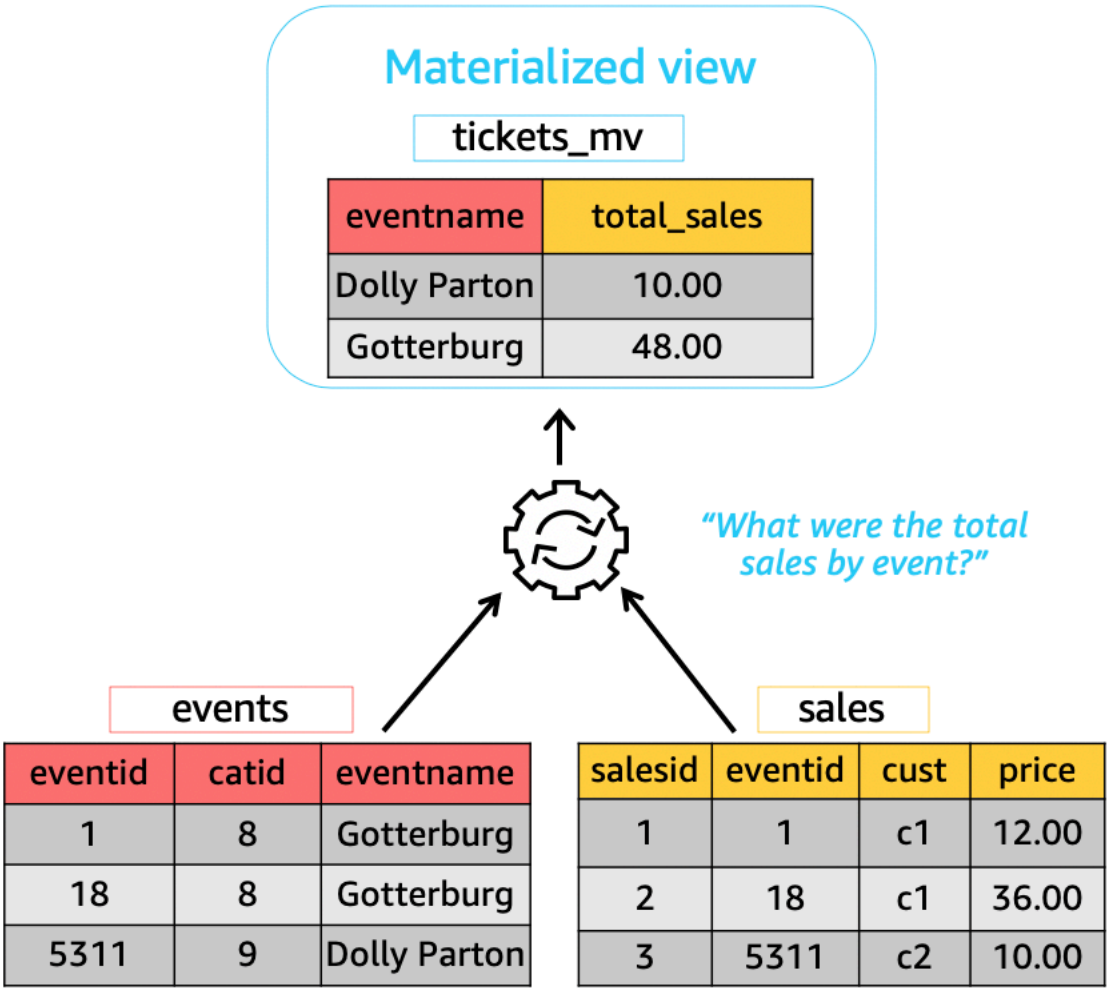
Les vues matérialisées sont particulièrement utiles pour accélérer les requêtes qui sont prévisibles et répétées. Au lieu d’effectuer des requêtes intensives en ressources sur des tables volumineuses (par exemple, des agrégations ou des jointures de plusieurs tables), les applications peuvent interroger une vue matérialisée et extraire un ensemble de résultats précalculés. Par exemple, considérez le scénario dans lequel un ensemble de requêtes est utilisé pour remplir des tableaux de bord, comme Amazon. QuickSight Ce cas d’utilisation est idéal pour une vue matérialisée, car les requêtes sont prévisibles et répétées indéfiniment.

Vous pouvez définir une vue matérialisée selon d’autres vues matérialisées. Utilisez les vues matérialisées sur des vues matérialisées pour étendre la capacité des vues matérialisées. Avec cette approche, une vue matérialisée existante joue le même rôle qu’une table de base afin que la requête puisse récupérer des données.

Cette approche est pertinente pour réutiliser des jointures précalculées pour différentes options d’agrégation ou GROUP BY. Par exemple, prenez une vue matérialisée qui joint les informations client (contenant des millions de lignes) aux informations détaillées sur la commande d’articles

(contenant des milliards de lignes). Il s'agit d'une requête coûteuse pouvant être lancée sur demande et de façon répétée. Vous pouvez utiliser différentes options GROUP BY pour les vues matérialisées créées au-dessus de cette vue matérialisée et les joindre à d'autres tables. Cela permet d'économiser du temps de calcul, autrement utilisé pour exécuter à chaque fois la jointure sous-jacente coûteuse. La table STV_MV_DEPS montre les dépendances d'une vue matérialisée sur d'autres vues matérialisées.

Lorsque vous créez une vue matérialisée, Amazon Redshift exécute l'instruction SQL définie par l'utilisateur pour collecter les données de la ou des tables de base et stocke l'ensemble de résultats. L'illustration suivante fournit une vue d'ensemble de la vue matérialisée `tickets_mv` définie par une requête SQL à l'aide de deux tables de base, `events` et `sales`.



Vous pouvez ensuite utiliser ces vues matérialisées dans les requêtes pour les accélérer. De plus, Amazon Redshift peut réécrire automatiquement ces requêtes pour utiliser des vues matérialisées, même lorsque la requête ne fait pas explicitement référence à une vue matérialisée. La réécriture

automatique des requêtes est particulièrement efficace afin d'améliorer les performances lorsque vous ne pouvez pas modifier vos requêtes pour utiliser des vues matérialisées.

Pour mettre à jour les données de la vue matérialisée, vous pouvez utiliser l'instruction REFRESH MATERIALIZED VIEW à tout moment afin d'actualiser manuellement les vues matérialisées. Amazon Redshift identifie les modifications qui ont eu lieu dans la ou les tables de base, puis applique ces modifications à la vue matérialisée. Étant donné que la réécriture automatique des requêtes nécessite que les vues matérialisées soient à jour, en tant que propriétaire d'une vue matérialisée, assurez-vous de l'actualiser à chaque changement d'une table de base.

Amazon Redshift donne quelques astuces pour maintenir les vues matérialisées à jour pour la réécriture automatique. Vous pouvez configurer des vues matérialisées avec l'option d'actualisation automatique afin d'actualiser les vues matérialisées lorsque leurs tables de base sont mises à jour. L'actualisation automatique s'exécute à un moment où les ressources du cluster sont disponibles pour minimiser les interruptions des autres applications. Étant donné que la planification de l'actualisation automatique dépend de la charge de travail, vous pouvez contrôler plus facilement le moment où Amazon Redshift actualise vos vues matérialisées. Vous pouvez planifier une tâche d'actualisation d'une vue matérialisée à l'aide de l'API du planificateur Amazon Redshift et de l'intégration de la console. Pour plus d'informations sur la planification des requêtes, consultez Planification d'une requête sur la console Amazon Redshift.

Cela est particulièrement utile lorsqu'un accord de niveau de service (SLA) est requis pour les up-to-date données issues d'une vue matérialisée. Toutes les vues matérialisées que vous actualisez automatiquement peuvent aussi être actualisées manuellement. Pour plus d'informations sur la création de vues matérialisées, consultez CREATE MATERIALIZED VIEW.

Vous pouvez émettre des instructions SELECT pour interroger une vue matérialisée. Pour plus d'informations sur l'interrogation des vues matérialisées, consultez Requêtes de vue matérialisées. L'ensemble de résultats finit par devenir obsolète à mesure que les données sont insérées, mises à jour et supprimées dans les tables de base. Vous pouvez actualiser la vue matérialisée à tout moment pour la mettre à jour avec les dernières modifications des tables de base. Pour plus d'informations sur l'actualisation des vues matérialisées, consultez REFRESH MATERIALIZED VIEW.

Pour plus d'informations sur les commandes SQL utilisées pour créer et gérer des vues matérialisées, consultez les rubriques suivantes :

- CREATE MATERIALIZED VIEW
- ALTER MATERIALIZED VIEW

- REFRESH MATERIALIZED VIEW
- DROP MATERIALIZED VIEW

Pour plus d'informations sur les tables et les vues système permettant de surveiller les vues matérialisées, consultez les rubriques suivantes :

- STV_MV_INFO
- STL_MV_STATE
- SVL_MV_REFRESH_STATUS
- STV_MV_DEPS

Rubriques

- Requêtes de vue matérialisées
- Réécriture automatique des requêtes pour utiliser des vues matérialisées
- Vues matérialisées sur des tables de lacs de données externes dans Amazon Redshift Spectrum
- Actualisation d'une vue matérialisée
- Vues matérialisées automatisées
- Utilisation d'une fonction définie par l'utilisateur (UDF) dans une vue matérialisée
- Diffusion de l'ingestion vers une vue matérialisée

Requêtes de vue matérialisées

Vos pouvez utiliser une vue matérialisée dans une requête SQL en référençant le nom de la vue matérialisée comme source de données, par exemple une table ou une vue standard.

Lorsqu'une requête accède à une vue matérialisée, elle ne voit que les données qui sont stockées dans la vue matérialisée actualisée la plus récemment. Par conséquent, la requête ne voit pas toujours toutes les dernières modifications des tables de base correspondantes de la vue matérialisée.

Si d'autres utilisateurs souhaitent interroger la vue matérialisée, le propriétaire de la vue matérialisée doit accorder l'autorisation SELECT à ces utilisateurs. Les autres utilisateurs n'ont pas besoin de l'autorisation SELECT sur les tables de base sous-jacentes. Le propriétaire de la vue matérialisée peut également révoquer l'autorisation SELECT pour les autres utilisateurs afin de les empêcher

d’interroger la vue matérialisée. Notez que les autres utilisateurs ont toujours besoin de l'autorisation USAGE sur les schémas contenant les tables de base de la vue matérialisée.

Si le propriétaire de la vue matérialisée ne dispose plus de l'autorisation SELECT locale sur les tables de base sous-jacentes :

- Le propriétaire ne peut plus interroger la vue matérialisée.
- Les autres utilisateurs qui disposent de l’autorisation SELECT sur la vue matérialisée ne peuvent plus interroger la vue matérialisée.

Cela est limité aux autorisations locales. Les modifications des autorisations gérées par Lake Formation ne sont pas vérifiées lors de l'interrogation de la vue matérialisée. Cela signifie que si une table de base sous-jacente est gérée par Lake Formation et que certaines autorisations sur la table sont révoquées dans Lake Formation, vous pouvez toujours interroger la vue matérialisée.

L’exemple suivant interroge la vue matérialisée `tickets_mv`. Pour plus d’informations sur la commande SQL utilisée pour créer une vue matérialisée, consultez CREATE MATERIALIZED VIEW.

```
SELECT sold
FROM tickets_mv
WHERE catgroup = 'Concerts';
```

Dans la mesure où les résultats de la requête sont précalculés, il n’est pas nécessaire d’accéder aux tables sous-jacentes (`category`, `event` et `sales`). Amazon Redshift peut renvoyer les résultats directement depuis `tickets_mv`.

Réécriture automatique des requêtes pour utiliser des vues matérialisées

Vous pouvez utiliser la réécriture automatique des requêtes des vues matérialisées pour qu’Amazon Redshift réécrive les requêtes afin d’utiliser les vues matérialisées. Cela accélère les applications des requêtes, même celles qui ne font pas explicitement référence à une vue matérialisée. Lorsqu’Amazon Redshift réécrit des requêtes, il utilise uniquement des vues matérialisées à jour.

Notes d’utilisation

Pour vérifier si la réécriture automatique des requêtes est utilisée, vérifiez le plan de requête ou STL_EXPLAIN. Voici une instruction SELECT et la sortie EXPLAIN du plan de requête d’origine.

```
SELECT catgroup, SUM(qtysold) AS sold
FROM category c, event e, sales s
WHERE c.catid = e.catid AND e.eventid = s.eventid
GROUP BY 1;

EXPLAIN
 XN HashAggregate  (cost=920021.24..920021.24 rows=1 width=35)
   ->  XN Hash Join DS_BCAST_INNER  (cost=440004.53..920021.22 rows=4 width=35)
         Hash Cond: ("outer".eventid = "inner".eventid)
         ->  XN Seq Scan on sales s  (cost=0.00..7.40 rows=740 width=6)
         ->  XN Hash  (cost=440004.52..440004.52 rows=1 width=37)
               ->  XN Hash Join DS_BCAST_INNER  (cost=0.01..440004.52 rows=1 width=37)
                     Hash Cond: ("outer".catid = "inner".catid)
                     ->  XN Seq Scan on event e  (cost=0.00..2.00 rows=200 width=6)
                     ->  XN Hash  (cost=0.01..0.01 rows=1 width=35)
                           ->  XN Seq Scan on category c  (cost=0.00..0.01 rows=1
 width=35)
```

Voici la sortie EXPLAIN après une réécriture automatique réussie. Cette sortie inclut une analyse de la vue matérialisée dans le plan de requête qui remplace des parties du plan de requête d’origine.

```
* EXPLAIN
     XN HashAggregate  (cost=11.85..12.35 rows=200 width=41)
       ->  XN Seq Scan on mv_tbl__tickets_mv__0 derived_table1  (cost=0.00..7.90
 rows=790 width=41)
```

Seules les up-to-date (nouvelles) vues matérialisées sont prises en compte pour la réécriture automatique des requêtes, quelle que soit la stratégie d'actualisation (automatique, planifiée ou manuelle). Par conséquent, la requête d'origine renvoie up-to-date des résultats. Lorsqu’une vue matérialisée est explicitement référencée dans les requêtes, Amazon Redshift accède aux données stockées dans la vue matérialisée. Il est possible que ces données ne reflètent pas les dernières modifications des tables de base de la vue matérialisée.

Vous pouvez utiliser la réécriture automatique des requêtes des vues matérialisées créées sur le cluster version 1.0.20949 ou ultérieure.

Vous pouvez interrompre la réécriture automatique des requêtes au niveau de la séance en définissant SET mv_enable_aqmv_for_session sur FALSE.

Comment fonctionne la réécriture automatique des requêtes des vues matérialisées

Sur la base d'une optimisation interne, Amazon Redshift peut décider d'invoquer la réécriture automatique des requêtes des vues matérialisées de manière transparente afin de fournir l'exécution des requêtes la plus optimale avec les temps de requête les plus courts possibles.

Supposons, par exemple, que l'utilisateur A crée une vue matérialisée M1 sur la table T1 avec la requête`SELECT * FROM T1`. L'utilisateur A dispose de tous les privilèges SELECT sur T1. L'utilisateur A accorde l'accès à M1 à tous les utilisateurs, y compris à l'utilisateur B. Cependant, lorsque l'utilisateur B essaie d'interroger directement T1, l'accès lui est refusé. Cela est dû au fait que l'utilisateur B ne dispose pas des privilèges SELECT sur T1.

Quelque temps plus tard, l'utilisateur B essaie à nouveau d'interroger T1, mais cette fois, il obtient les résultats de T1. Cela est dû au fait que la réécriture automatique des requêtes à l'aide de la vue matérialisée a réécrit la requête de l'utilisateur B sur la table T1 (`SELECT <cols> FROM T1`) en une requête sur la vue matérialisée M1 (). `SELECT <cols> FROM M1`

Limites

Voici les limites de l’utilisation de la réécriture automatique des requêtes des vues matérialisées :

- La réécriture automatique des requêtes fonctionne avec des vues matérialisées qui ne font pas référence à ou n’incluent aucun des éléments suivants :
  - Sous-requêtes
  - Outer join left, right ou full
  - Opérations d’ensemble
  - Toutes les fonctions d’agrégation, sauf SUM, COUNT, MIN, MAX et AVG. (Il s’agit des seules fonctions d’agrégation qui fonctionnent avec la réécriture automatique des requêtes.)
  - Toutes les fonctions d’agrégation avec DISTINCT
  - Toutes les fonctions de fenêtrage
  - Clauses SELECT DISTINCT ou HAVING
  - Tables externes ou partagées
  - Toute autre vue matérialisée

- La réécriture automatique des requêtes réécrit les requêtes SELECT qui font référence aux tables Amazon Redshift définies par l'utilisateur. Amazon Redshift ne réécrit pas les requêtes suivantes :
  - Les instructions CREATE TABLE AS
  - Les instructions SELECT INTO
  - Les requêtes sur des catalogues ou des tables système
  - Les requêtes avec jointures externes ou clause SELECT DISTINCT
- Si une requête n'est pas réécrite automatiquement, vérifiez si vous disposez de l'autorisations SELECT sur la vue matérialisée et si l'option mv_enable_aqmv_for_session est définie sur TRUE.

  Vous pouvez également vérifier si vos vues matérialisées sont éligibles à la réécriture automatique des requêtes en inspectant STV_MV_INFO. Pour de plus amples informations, veuillez consulter STV_MV_INFO.

Vues matérialisées sur des tables de lacs de données externes dans Amazon Redshift Spectrum

Les vues matérialisées peuvent fournir une maintenance incrémentielle sur les tables de lacs de données externes. Grâce à la maintenance incrémentielle, Amazon Redshift met à jour les données de la vue matérialisée avec uniquement les modifications apportées aux données des tables de base depuis la dernière actualisation. La maintenance incrémentielle est plus rentable que le recalcul complet de la vue matérialisée après chaque modification des données de la table de base.

Lorsque vous utilisez des vues matérialisées sur au moins une table externe, la création de vues matérialisées est incrémentielle sur :

- Tableaux de lacs de données standard, partitionnés et non partitionnés, avec des fichiers de données dans tous les formats pris en charge (Parquet, Avro, CSV, etc.).
- Tables Apache Iceberg, partitionnées et non partitionnées, avec et. copy-on-write merge-on-read
- Les tables Amazon Redshift Spectrum sont jointes à n'importe quelle table Amazon Redshift de la même base de données.

L'actualisation de la vue matérialisée est incrémentielle dans les cas suivants :

- Tables de lac de données standard après le remplacement de S3 DELETE ou PUT (suppression de fichiers de données), si la vue matérialisée n'effectue pas d'agrégation.

- Tables Apache Iceberg après le compactage INSERT, DELETE, UPDATE ou après compactage des tables.

Pour plus d'informations sur Amazon Redshift Spectrum, Amazon Redshift Spectrum consultez.

Limites

Les limitations générales relatives aux vues matérialisées s'appliquent toujours aux vues matérialisées sur les tables des lacs de données. Pour de plus amples informations, veuillez consulter Actualisation d'une vue matérialisée. En outre, tenez compte des limites suivantes lorsque vous utilisez des vues matérialisées sur des tables de lacs de données externes.

- La création de vues matérialisées n'est pas incrémentielle dans les cas suivants :
  - Tables Hudi ou Delta Lake.
  - Accès aux données imbriqué dans le spectre.
  - Références aux colonnes VARBYTE.
- L'actualisation de la vue matérialisée revient à un recalcul complet sur :
  - Apache Iceberg établit des tables lorsqu'un instantané requis a expiré, si la vue matérialisée effectue une agrégation.
  - Tables de lac de données standard après suppression ou mise à jour de fichiers de données sur Amazon S3, si la vue matérialisée effectue une agrégation.
  - Les tables de data lake standard ont été actualisées plusieurs fois au sein d'un bloc de transactions.
  - Tableaux de lacs de données standard régis par un manifeste. Pour plus d'informations sur les manifestes, consultez la section Utilisation d'un manifeste pour spécifier des fichiers de données.
  - Amazon Redshift revient au recalcul complet s'il est censé être plus performant, en particulier pour les vues matérialisées contenant des jointures et si plusieurs tables de base ont été mises à jour depuis la dernière actualisation.
- Sur les tables Apache Iceberg, l'actualisation des vues matérialisées ne peut gérer que 4 millions de positions supprimées dans un seul fichier de données. Une fois cette limite atteinte, la table de base d'Apache Iceberg doit être compactée pour continuer à actualiser la vue matérialisée.
- Sur les tables Apache Iceberg, le dimensionnement de la simultanéité n'est pas pris en charge pour la création et l'actualisation de vues matérialisées.

- Les fonctionnalités autonomes ne sont pas prises en charge. Il s'agit notamment des vues matérialisées automatisées, de l'actualisation automatique et de la réécriture automatique des requêtes.
- Lorsqu'une vue matérialisée incrémentielle est actualisée, les autorisations IAM s'appliquent uniquement aux parties consultées des tables de base Amazon Redshift.
- Les modifications des autorisations gérées par Lake Formation ne sont pas vérifiées lors de l'interrogation d'une vue matérialisée. Cela signifie que si une vue matérialisée est définie sur une table de lac de données et que les privilèges de sélection sont supprimés de la table avec Lake Formation, vous pouvez toujours interroger la vue matérialisée.

Actualisation d’une vue matérialisée

Cette rubrique décrit comment actualiser les données d'une vue matérielle à partir des tables sous-jacentes.

Lorsque vous créez une vue matérialisée, son contenu reflète l’état de la ou des tables de base de données sous-jacentes à ce moment-là. Les données de la vue matérialisée restent inchangées, même si les applications modifient les données dans les tables sous-jacentes. Pour mettre à jour les données de la vue matérialisée, vous pouvez utiliser l’instruction `REFRESH MATERIALIZED VIEW` à tout moment afin d’actualiser manuellement les vues matérialisées. Lorsque vous utilisez cette instruction, Amazon Redshift identifie les modifications qui ont eu lieu dans la ou les tables de base, puis applique ces modifications à la vue matérialisée.

Amazon Redshift offre deux politiques pour actualiser une vue matérialisée :

- Dans de nombreux cas, Amazon Redshift peut effectuer une actualisation incrémentielle. Dans une actualisation incrémentielle, Amazon Redshift identifie rapidement les modifications apportées aux données dans les tables de base depuis la dernière actualisation et met à jour les données dans la vue matérialisée. L’actualisation incrémentielle est prise en charge sur les constructions SQL suivantes utilisées dans la requête lors de la définition de la vue matérialisée :
  - Les constructions qui contiennent les clauses SELECT, FROM, [INNER] JOIN, WHERE, GROUP BY ou HAVING.
  - Les constructions qui contiennent des agrégations, telles que SUM, MIN, MAX, AVG et COUNT.
  - La plupart des fonctions SQL intégrées, en particulier celles qui sont immuables, étant donné qu’elles ont les mêmes arguments d’entrée et produisent toujours la même sortie.

L’actualisation incrémentielle est également prise en charge pour une vue matérialisée basée sur une table d’unité de partage des données.

- Si une actualisation incrémentielle est impossible, Amazon Redshift effectue une actualisation complète. Une actualisation complète réexécute l’instruction SQL sous-jacente, en remplaçant toutes les données de la vue matérialisée.
- Amazon Redshift choisit automatiquement la méthode d'actualisation pour une vue matérialisée en fonction de la requête SELECT utilisée pour définir la vue matérialisée.

L’actualisation d’une vue matérialisée sur une vue matérialisée n’est pas un processus en cascade. En d'autres termes, supposons que vous disposiez d'une vue matérialisée A qui dépend de la vue matérialisée B. Dans ce cas, lorsque le REFRESH MATERIALIZED VIEW A est invoqué, A est actualisé à l'aide de la version actuelle de B, même si B l'est. out-of-date Pour mettre la vue A à jour, actualisez d’abord la vue B dans une transaction distincte.

L’exemple suivant montre comment créer un plan d’actualisation complet pour une vue matérialisée par programmation. Pour actualiser la vue matérialisée v, actualisez d’abord la vue matérialisée u. Pour actualiser la vue matérialisée w, actualisez d’abord la vue matérialisée u, puis la vue matérialisée v.

```
CREATE TABLE t(a INT);
CREATE MATERIALIZED VIEW u AS SELECT * FROM t;
CREATE MATERIALIZED VIEW v AS SELECT * FROM u;
CREATE MATERIALIZED VIEW w AS SELECT * FROM v;

WITH RECURSIVE recursive_deps (mv_tgt, lvl, mv_dep) AS
( SELECT trim(name) as mv_tgt, 0 as lvl, trim(ref_name) as mv_dep
  FROM stv_mv_deps
  UNION ALL
  SELECT R.mv_tgt, R.lvl+1 as lvl, trim(S.ref_name) as mv_dep
  FROM stv_mv_deps S, recursive_deps R
  WHERE R.mv_dep = S.name
)

SELECT mv_tgt, mv_dep from recursive_deps
ORDER BY mv_tgt, lvl DESC;

 mv_tgt | mv_dep
--------+--------
 v      | u
```

```
 w      | u
 w      | v
(3 rows)
```

L'exemple suivant affiche un message d'information lorsque vous exécutez REFRESH MATERIALIZED VIEW sur une vue matérialisée qui dépend d'une vue out-of-date matérialisée.

```
create table a(a int);
```

```
create materialized view b as select * from a;
```

```
create materialized view c as select * from b;
```

```
insert into a values (1);
```

```
refresh materialized view c;

INFO:  Materialized view c is already up to date.  However, it depends on another
 materialized view that is not up to date.
```

```
REFRESH MATERIALIZED VIEW b;
INFO:  Materialized view b was incrementally updated successfully.
```

```
REFRESH MATERIALIZED VIEW c;
INFO:  Materialized view c was incrementally updated successfully.
```

Pour l’instant, Amazon Redshift présente les limitations suivantes pour l’actualisation incrémentielle des vues matérialisées.

Amazon Redshift ne prend pas en charge l’actualisation incrémentielle pour les vues matérialisées qui sont définies avec une requête utilisant les éléments SQL suivants :

- OUTER JOIN (RIGHT, LEFT ou FULL).
- Les opérations d’ensemble UNION, INTERSECT, EXCEPT et MINUS.
- Les fonctions d’agrégation MEDIAN, PERCENTILE_CONT, LISTAGG, STDDEV_SAMP, STDDEV_POP, APPROXIMATE COUNT, APPROXIMATE PERCENTILE et les fonctions d’agrégation bit par bit.

> **Note**
>
> Les fonctions d'agrégation COUNT, SUM et AVG sont prises en charge.

- Fonctions d'agrégation DISTINCT, telles que DISTINCT COUNT, DISTINCT SUM, etc.
- Fonctions de fenêtrage.
- Requête qui utilise des tables temporaires pour l'optimisation des requêtes, telle que l'optimisation des sous-expressions courantes.
- Sous-requêtes.
- Tables externes référençant les formats suivants dans la requête qui définit la vue matérialisée.
  - Delta Lake
  - Hudi

  L'actualisation incrémentielle est prise en charge pour les vues matérialisées définies à l'aide de formats autres que ceux répertoriés ci-dessus. Pour de plus amples informations, veuillez consulter Vues matérialisées sur des tables de lacs de données externes dans Amazon Redshift Spectrum.

Actualisation automatique d'une vue matérialisée

Amazon Redshift peut actualiser automatiquement les vues matérialisées avec les up-to-date données de ses tables de base lorsque des vues matérialisées sont créées ou modifiées pour inclure l'option d'actualisation automatique. Amazon Redshift actualise automatiquement les vues matérialisées dès que possible après la modification des tables de base.

Pour terminer l'actualisation des vues matérialisées les plus importantes avec un impact minimal sur les applications actives dans votre cluster, Amazon Redshift prend en compte plusieurs facteurs. Ces facteurs incluent la charge système actuelle, les ressources nécessaires à l'actualisation, les ressources de cluster disponibles et la fréquence d'utilisation des vues matérialisées.

Amazon Redshift privilégie vos charges de travail par rapport à l'actualisation automatique et peut arrêter cette dernière pour préserver les performances de la charge de travail des utilisateurs. Cette approche peut retarder l'actualisation de certaines vues matérialisées. Dans certains cas, il se peut que vous ayez besoin d'un comportement d'actualisation plus déterministe pour vos vues matérialisées. Le cas échéant, envisagez une actualisation manuelle comme décrit dans REFRESH MATERIALIZED VIEW ou une actualisation planifiée à l'aide des opérations d'API du planificateur Amazon Redshift ou de la console.

Vous pouvez définir l'actualisation automatique des vues matérialisées à l'aide de CREATE MATERIALIZED VIEW. Vous pouvez également utiliser la clause AUTO REFRESH pour actualiser automatiquement les vues matérialisées. Pour plus d'informations sur la création de vues matérialisées, consultez CREATE MATERIALIZED VIEW. Vous pouvez activer l'actualisation automatique d'une vue matérialisée via ALTER MATERIALIZED VIEW.

Tenez compte des éléments suivants lorsque vous actualisez des vues matérialisées :

- Vous pouvez toujours actualiser directement une vue matérialisée à l'aide de la commande REFRESH MATERIALIZED VIEW même si vous n'avez pas activé l'actualisation automatique.
- Amazon Redshift n'actualise pas automatiquement les vues matérialisées définies sur les tables externes.
- Pour en savoir plus sur le statut d'actualisation, vous pouvez vérifier SVL_MV_REFRESH_STATUS, qui enregistre les requêtes lancées par l'utilisateur ou actualisées automatiquement.
- Pour exécuter REFRESH sur des vues matérialisées de recalcul, assurez-vous de disposer de l'autorisation CREATE sur les schémas. Pour de plus amples informations, veuillez consulter GRANT.

Vues matérialisées automatisées

Cette rubrique décrit comment Amazon Redshift utilise des vues matérialisées automatisées pour améliorer les performances. Amazon Redshift crée automatiquement des vues matérialisées en fonction de l'activité et des performances de la base de données. Amazon Redshift utilise des vues matérialisées automatisées par défaut.

Les vues matérialisées sont un outil performant pour améliorer les performances des requêtes dans Amazon Redshift. Pour ce faire, elles stockent un ensemble de résultats précalculé. Les requêtes similaires n'ont pas besoin de réexécuter la même logique à chaque fois, car elles peuvent récupérer les enregistrements de l'ensemble de résultats existant. Les développeurs et les analystes créent des vues matérialisées après avoir analysé leurs charges de travail afin de déterminer quelles requêtes seraient bénéfiques et si le coût de maintenance de chaque vue matérialisée en vaut la peine. À mesure que les charges de travail augmentent ou changent, ces vues matérialisées doivent être examinées pour s'assurer qu'elles continuent d'offrir des avantages tangibles en matière de performances.

La fonctionnalité Automated Materialized Views (AutoMV) de Redshift améliore les performances des requêtes en créant et en gérant automatiquement des vues matérialisées basées sur la surveillance de la charge de travail et des algorithmes d'apprentissage automatique. Les principales fonctionnalités d'AutoMV sont les suivantes :

- Surveillance continue — Redshift surveille en permanence la charge de travail à l'aide de techniques d'apprentissage automatique afin d'identifier les opportunités d'amélioration des performances grâce à la création de vues matérialisées.
- Création et suppression automatiques : lorsque le système détecte qu'une vue matérialisée serait bénéfique, il la crée et la gère automatiquement. À l'inverse, si un AutoMV créé précédemment n'offre plus d'avantages en termes de performances, le système le supprimera automatiquement.
- Aucune activité utilisateur requise : la fonctionnalité AutoMV ne fonctionne que pendant les périodes de faible activité utilisateur ou de faible charge de travail sur le cluster. Cela garantit que les opérations AutoMV n'interfèrent pas avec les charges de travail des clients ni n'ont d'impact sur elles.
- Pics d'utilisation du processeur — En période d'absence de charge de travail, la création ou l'actualisation de vues matérialisées par AutoMV peut entraîner des pics d'utilisation du processeur. Il s'agit d'un comportement normal car le système utilise les ressources disponibles pour créer et actualiser des vues matérialisées.
- Priorité de la charge de travail utilisateur : si vous lancez une charge de travail alors qu'une opération AutoMV est en cours, la tâche AutoMV s'arrête pour libérer des ressources pour la charge de travail de l'utilisateur. Cela garantit que vos charges de travail ont la priorité sur les opérations AutoMV.

Bien que la fonction AutoMV puisse entraîner des pics d'utilisation du processeur pendant les périodes d'inactivité de l'utilisateur, elle fonctionne de manière transparente et sans impact sur vos charges de travail. Le système gère les vues matérialisées pour améliorer les performances des requêtes et hiérarchise simultanément les charges de travail des utilisateurs par rapport aux opérations AutoMV.

Le comportement et les capacités d’AutoMV sont les mêmes que les vues matérialisées créées par l’utilisateur. Elles sont actualisées automatiquement et progressivement, en utilisant les mêmes critères et restrictions. Tout comme les vues matérialisées créées par les utilisateurs, Réécriture automatique des requêtes pour utiliser des vues matérialisées identifie les requêtes qui peuvent bénéficier de la fonction Auto créée par le système. MVs Il réécrit automatiquement ces requêtes

pour utiliser l'AutoMVs, améliorant ainsi les performances des requêtes. Les développeurs n'ont pas besoin de corriger les requêtes pour tirer parti d'AutoMV.

> **Note**
>
> Les vues matérialisées automatisées sont actualisées par intermittence. Les requêtes réécrites pour utiliser AutoMV renvoient toujours les derniers résultats. Lorsque Redshift détecte que les données ne sont pas à jour, les requêtes ne sont pas réécrites pour être lues à partir de vues matérialisées automatisées. Les requêtes sélectionnent plutôt les données les plus récentes à partir des tables de base.

Toute charge de travail comportant des requêtes utilisées à plusieurs reprises peut bénéficier d'AutoMV. Cas d'utilisation courants :

- Tableaux de bord - Les tableaux de bord sont largement utilisés pour fournir un aperçu rapide des indicateurs commerciaux clés (KPIs), des événements, des tendances et d'autres indicateurs. Ils ont souvent une disposition commune avec des diagrammes et des tableaux, mais présentent des vues différentes pour le filtrage ou pour les opérations de sélection de dimensions, comme l'exploration vers le bas. Les tableaux de bord ont souvent un ensemble commun de requêtes utilisées à plusieurs reprises avec différents paramètres. Les requêtes de tableau de bord peuvent bénéficier grandement des vues matérialisées automatisées.
- Rapports : les requêtes de rapport peuvent être planifiées à diverses fréquences, en fonction des besoins métier et du type de rapport. En outre, elles peuvent être automatisées ou à la demande. L'une des caractéristiques courantes des requêtes de rapport est qu'elles peuvent s'avérer longues et gourmandes en ressources. Avec AutoMV, ces requêtes n'ont pas besoin d'être recalculées à chacune de leur exécution, ce qui réduit le temps d'exécution pour chaque requête ainsi que l'utilisation de ressources dans Redshift.

Pour désactiver les vues matérialisées automatisées, vous mettez à jour le groupe de paramètres `auto_mv` sur `false`. Pour plus d'informations, consultez Groupes de paramètres Amazon Redshift dans le Guide de gestion du cluster Amazon Redshift.

Portée et considérations SQL pour les vues matérialisées automatisées

- Une vue matérialisée automatisée peut être déclenchée et créée par une requête ou une sous-requête, à condition qu'elle contienne une clause `GROUP BY` ou l'une des fonctions d'agrégation

suivantes : SUM, COUNT, MIN, MAX ou AVG. Cependant, elle ne peut contenir aucun des éléments suivants :

- Outer join left, right ou full
- Fonctions d'agrégation autres que SUM, COUNT, MIN, MAX et AVG. (Ces fonctions particulières font appel à la réécriture automatique des requêtes).
- Toute fonction d'agrégation qui inclut DISTINCT
- Toutes les fonctions de fenêtrage
- Clauses SELECT DISTINCT ou HAVING
- Toute autre vue matérialisée

Une requête qui répond aux critères ne lancera pas nécessairement la création d'une vue matérialisée automatisée. Le système détermine les candidats à partir desquels créer une vue, en fonction de l'avantage escompté pour la charge de travail et du coût en ressources à maintenir, ce qui inclut le coût d'actualisation du système. Chaque vue matérialisée qui en résulte est utilisable par la réécriture automatique de requêtes.

- Même si AutoMV peut être lancé par le biais d'une sous-requête ou par des tronçons individuels d'opérateurs d'ensembles, la vue matérialisée qui en résulte ne contiendra pas de sous-requêtes ou d'opérateurs d'ensemble.
- Pour déterminer si AutoMV a été utilisé pour les requêtes, consultez le plan EXPLAIN et cherchez l'élément `%_auto_mv_%` dans la sortie. Pour obtenir plus d'informations, consultez EXPLAIN.
- Les vues matérialisées automatisées ne sont pas prises en charge sur les tables externes, telles que les unités de partage des données et les tables fédérées.

Vues matérialisées automatisées

Voici les limites de l'utilisation des vues matérialisées automatisées :

- Nombre maximum de vues automatiques MVs : la limite des vues matérialisées automatisées est de 200 par base de données du cluster.
- Storage space and capacity (Espace et capacité de stockage) : une caractéristique importante d'AutoMV est qu'il s'exécute à l'aide de cycles en arrière-plan de secours pour aider à ce que les charges de travail des utilisateurs ne soient pas affectées. Si le cluster est occupé ou manque d'espace de stockage, AutoMV interrompt son activité. Plus précisément, à 80 % de la capacité totale du cluster, aucune nouvelle vue matérialisée automatisée n'est créée. À 90 % de la capacité totale, elles peuvent être supprimées afin que les charges de travail des utilisateurs se poursuivent

sans dégradation des performances. Pour plus d'informations sur comment déterminer la capacité de cluster, consultez STV_NODE_STORAGE_CAPACITY.

Facturation pour les vues matérialisées automatisées

La fonctionnalité d'optimisation automatique d'Amazon Redshift crée et actualise les vues matérialisées automatisées. Les ressources de calcul utilisées dans le cadre de ce processus n'entraînent pas de frais. Le stockage des vues matérialisées automatisées est facturé au tarif normal du stockage. Pour plus d'informations, consultez Tarification d'Amazon Redshift.

Ressources supplémentaires

Le billet de blog suivant fournit des explications supplémentaires concernant les vues matérialisées automatisées, notamment sur la façon de les créer, de les gérer et de les supprimer. Il décrit également les algorithmes sous-jacents qui sous-tendent ces décisions : Optimize your Amazon Redshift query performance with automated materialized views.

Cette vidéo commence par une description des vues matérialisées et montre en quoi elles améliorent les performances et préservent les ressources. Elle décrit ensuite de manière approfondie les vues matérialisées automatisées avec une animation de flux de processus et une démonstration en direct.

Utilisation d'une fonction définie par l'utilisateur (UDF) dans une vue matérialisée

Vous pouvez utiliser un UDF scalaire dans une vue matérialisée Amazon Redshift. Définissez-les en python ou en SQL et faites-y référence dans la définition de la vue matérialisée.

Référencer un UDF dans une vue matérialisée

La procédure suivante montre comment effectuer des comparaisons UDFs arithmétiques simples, dans une définition de vue matérialisée.

1. Créez une table à utiliser dans la définition de la vue matérialisée.

```
CREATE TABLE base_table (a int, b int);
```

2. Créez une fonction scalaire définie par l'utilisateur en python qui renvoie une valeur booléenne indiquant si un entier est plus grand qu'un entier de comparaison.

```
CREATE OR REPLACE FUNCTION udf_python_bool(x1 int, x2 int) RETURNS bool IMMUTABLE
AS $$
  return x1 > x2
$$ LANGUAGE plpythonu;
```

Si vous le souhaitez, vous pouvez créer un UDF fonctionnellement similaire avec SQL, que vous pouvez utiliser pour comparer les résultats avec le premier.

```
CREATE OR REPLACE FUNCTION udf_sql_bool(int, int) RETURNS bool IMMUTABLE
AS $$
  select $1 > $2;
$$ LANGUAGE SQL;
```

3. Créez une vue matérialisée qui sélectionne dans la table que vous avez créée et qui fait référence à l'UDF.

```
CREATE MATERIALIZED VIEW mv_python_udf AS SELECT udf_python_bool(a, b) AS a FROM
 base_table;
```

En option, vous pouvez créer une vue matérialisée qui fait référence à l'UDF SQL.

```
CREATE MATERIALIZED VIEW mv_sql_udf AS SELECT udf_sql_bool(a, b) AS a FROM
 base_table;
```

4. Ajoutez des données à la table et actualisez la vue matérialisée.

```
INSERT INTO base_table VALUES (1,2), (1,3), (4,2);
```

```
REFRESH MATERIALIZED VIEW mv_python_udf;
```

En option, vous pouvez actualiser la vue matérialisée qui fait référence à l'UDF SQL.

```
REFRESH MATERIALIZED VIEW mv_sql_udf;
```

5. Interrogez les données de votre vue matérialisée.

```
SELECT * FROM mv_python_udf ORDER BY a;
```

Les résultats de la requête sont les suivants :

```
a
-----
false
false
true
```

Le résultat est `true` pour la dernière série de valeurs car la valeur de la colonne a (4) est supérieure à la valeur de la colonne b (2).

6. En option, vous pouvez interroger la vue matérialisée qui référence l'UDF SQL. Les résultats de la fonction SQL correspondent aux résultats de la version Python.

```
SELECT * FROM mv_sql_udf ORDER BY a;
```

Les résultats de la requête sont les suivants :

```
a
-----
false
false
true
```

Cette fonction renvoie `true` pour la dernière série de valeurs à comparer.

7. Utilisez une instruction DROP avec CASCADE pour supprimer la fonction définie par l'utilisateur et la vue matérialisée qui y fait référence.

```
DROP FUNCTION udf_python_bool(int, int) CASCADE;
```

```
DROP FUNCTION udf_sql_bool(int, int) CASCADE;
```

Diffusion de l'ingestion vers une vue matérialisée

Cette rubrique explique comment utiliser les vues matérialisées pour accéder rapidement aux données de streaming.

L'ingestion de streaming permet une ingestion de données à faible latence et à haut débit depuis Amazon Kinesis Data Streams ou Amazon Managed Streaming for Apache Kafka vers une base de données Amazon Redshift provisionnée ou Amazon Redshift Serverless. Les données arrivent dans une vue matérialisée Redshift configurée à cet effet. Cela se traduit par un accès rapide aux données externes. L'ingestion du streaming réduit le temps d'accès aux données et réduit les coûts de stockage. Vous pouvez le configurer pour votre cluster Amazon Redshift ou pour votre groupe de travail Amazon Redshift Serverless à l'aide d'un petit ensemble de commandes SQL. Une fois configurée, chaque actualisation d'une vue matérialisée peut ingérer des centaines de mégaoctets de données par seconde.

Comment les données circulent d'un service de streaming vers Redshift

Cela permet de comprendre le fonctionnement de l'ingestion du streaming et les objets de base de données utilisés au cours du processus. Les données circulent directement d'un fournisseur de flux de données vers un cluster provisionné par Amazon Redshift ou vers un groupe de travail Amazon Redshift Serverless. Il n'existe pas de zone d'atterrissage temporaire, telle qu'un compartiment Amazon S3. Le cluster ou le groupe de travail provisionné est le consommateur du flux. Dans la base de données Redshift, les données lues dans le flux apparaissent dans une vue matérialisée. Les données sont traitées au fur et à mesure de leur arrivée. Par exemple, les valeurs JSON peuvent être consommées et mappées aux colonnes de données d'une vue matérialisée à l'aide de SQL. Lorsque la vue matérialisée est actualisée, Redshift consomme les données provenant des fragments de données Kinesis ou des partitions Kafka alloués jusqu'à ce que la vue soit mise à jour avec le flux.

Les cas d'utilisation de l'ingestion du streaming par Amazon Redshift impliquent des données générées en continu et devant être traitées dans un court laps de temps (latence) à compter de leur origine. C'est ce que l'on appelle communément l'analyse en temps quasi réel. Les sources peuvent inclure des appareils informatiques, des dispositifs de télémétrie du système et des données de flux de clics provenant d'un site Web ou d'une application très fréquenté.

Meilleures pratiques d'analyse des données pour améliorer les performances

Lorsque vous configurez l'ingestion du streaming, il existe des options permettant d'analyser les données entrantes. Les pratiques peuvent inclure l'exécution de la logique métier ou le formatage au fur et à mesure que les données arrivent. Nous recommandons les meilleures pratiques suivantes pour éviter les erreurs ou les pertes de données. Ils sont issus de tests internes et ont aidé les clients à résoudre les problèmes de configuration et d'analyse.

- Extraction de valeurs à partir de données diffusées : si vous utilisez la fonction JSON_EXTRACT_PATH_TEXT dans la définition de votre vue matérialisée pour analyser ou détruire le JSON diffusé, cela peut avoir un impact significatif sur les performances et la latence. Pour expliquer, pour chaque colonne extraite à l'aide de JSON_EXTRACT_PATH_TEXT, le JSON entrant est réanalysé. Ensuite, la conversion des types de données, le filtrage et les calculs de logique métier ont lieu. Cela signifie, par exemple, que si vous extrayez 10 colonnes à partir de données JSON, chaque enregistrement JSON est analysé 10 fois, ce qui inclut une logique supplémentaire. Cela se traduit par une latence d'ingestion plus élevée. Une autre approche que nous recommandons consiste à utiliser la fonction JSON_PARSE pour convertir les enregistrements JSON au type de données SUPER de Redshift. Une fois que les données diffusées ont atterri dans la vue matérialisée, utilisez partiQL pour extraire des chaînes individuelles de la représentation SUPER des données JSON. Pour plus d'informations, consultez la section Interrogation de données semi-structurées.

  Notez également que JSON_EXTRACT_PATH_TEXT a une taille de données maximale de 64 Ko. Ainsi, si un enregistrement JSON est supérieur à 64 Ko, son traitement avec JSON_EXTRACT_PATH_TEXT entraîne une erreur.

- Mappage d'un Amazon Kinesis Data Streams flux ou d'une rubrique Amazon MSK sur plusieurs vues matérialisées : nous vous déconseillons de créer plusieurs vues matérialisées pour ingérer les données d'un seul flux ou sujet. En effet, chaque vue matérialisée crée un consommateur pour chaque partition du flux ou de la partition Kinesis Data Streams de la rubrique Kafka. Cela peut entraîner une limitation ou un dépassement du débit du flux ou du sujet. Cela peut également entraîner des coûts plus élevés, car vous ingérez les mêmes données plusieurs fois. Lorsque vous configurez l'ingestion de flux, nous vous recommandons de créer une vue matérialisée pour chaque flux ou sujet.

  Si votre cas d'utilisation nécessite que vous ingériez les données d'un flux KDS ou d'un sujet MSK dans plusieurs vues matérialisées, consultez le blog AWS Big Data, en particulier les meilleures pratiques pour implémenter des analyses à l'aide d' near-real-timeAmazon Redshift Streaming Ingestion with Amazon MSK, avant de le faire.

Comportement d'ingestion du streaming et types de données

Le tableau suivant décrit les détails techniques du comportement et les limites de taille pour différents types de données. Nous vous recommandons de vous familiariser avec celles-ci avant de configurer une vue matérialisée pour l'ingestion du streaming.

Fonction ou comportement	Description
Kafka topic length limit (Limite de longueur des rubriques Kafka)	Il n'est pas possible d'utiliser une rubrique Kafka dont le nom comporte plus de 128 caractères (guillemets non compris). Pour plus d'informations, consultez Noms et identificateurs.
Actualisations incrémentielles et JOINs sur une vue matérialisée	La vue matérialisée doit être gérable de manière progressive. Le recalcul complet est impossible pour Kinesis ou Amazon MSK, car ils ne conservent pas l'historique des flux ou des rubriques après 24 heures ou 7 jours, par défaut. Vous pouvez définir des périodes de conservation des données plus longues dans Kinesis ou Amazon MSK. Cependant, cela peut entraîner une maintenance et des frais supplémentaires. En outre, JOINs ils ne sont actuellement pas pris en charge sur les vues matérialisées créées sur un flux Kinesis ou sur une rubrique Amazon MSK. Après avoir créé une vue matérialisée sur votre flux ou rubrique, vous pouvez créer une autre vue matérialisée pour joindre votre vue matérialisée de streaming à d'autres vues matérialisées, tables ou vues. Pour plus d'informations, consultez REFRESH MATERIALIZED VIEW.
Record parsing (Analyse des enregistrements)	L'ingestion en streaming Amazon Redshift ne prend pas en charge l'analyse des enregistrements agrégés par la Kinesis Producer Library (KPL Key Concepts - Aggregation (Concepts clés KPL – Agrégation)). Les registres agrégés sont ingérés, mais sont stockés sous forme de données de tampon de protocole binaire. (Consultez Protocol Buffers pour plus d'informations.) Selon la manière dont vous transmettez les données à Kinesis, vous devrez peut-être désactiver cette fonction.
Valeurs dupliquées dans les en-têtes Kafka	Le client de streaming grand public Amazon Redshift pour les sujets Kafka provenant d'Amazon MSK, Confluent ou Apache Kafka ne prend pas en charge les valeurs dupliquées dans les en-têtes des rubriques Kafka.

Fonction ou comportement	Description
Decompression (Décompression)	VARBYTEne prend pas en charge la décompression. De ce fait, les enregistrements contenant des données compressées ne peuvent pas être interrogés dans Redshift. Décompressez vos données avant de les ajouter au flux Kinesis ou à la rubrique Amazon MSK.
Maximum record size (Taille d'enregistrement maximale)	La taille maximale de tout enregistrement qu'Amazon Redshift peut ingérer depuis Kinesis ou Amazon MSK est de 16 777 216 octets (16 MiB), taille maximale prise en charge par le type de données VARBYTE dans Amazon Redshift. Par défaut, les vues matérialisées en streaming Amazon Redshift créées sur un flux de données Kinesis ou une rubrique Amazon MSK fixent la taille de la colonne de données à 1 048 576 octets (1 Mo) et 16 777 216 octets (16 Mo) respectivement. **Note** 1 Mo est la taille maximale actuelle de tout enregistrement pouvant être placé dans un flux de données Kinesis. Pour plus d'informations sur les limites de taille Kinesis, consultez la section Quotas et limites du manuel Amazon Kinesis Data Streams Developer Guide.
Enregistrements d'erreurs	Chaque fois qu'un enregistrement ne peut pas être ingéré dans Redshift parce que la taille des données dépasse le maximum, cet enregistrement est ignoré. L'actualisation de la vue matérialisée réussit toujours, dans ce cas, et un segment de chaque enregistrement d'erreur est écrit dans la table système SYS_STREAM_SCAN_ERRORS. Les erreurs résultant de la logique métier, telles qu'une erreur de calcul ou une erreur résultant d'une conversion de type, ne sont pas ignorées. Testez soigneusement la logique avant de l'ajouter à la définition de votre vue matérialisée.

Fonction ou comportement	Description
Connectivité privée multi-VPC Amazon MSK	La connectivité privée multi-VPC d'Amazon MSK n'est actuellement pas prise en charge pour l'ingestion du streaming Redshift. Vous pouvez également utiliser le peering VPC pour connecter VPCs ou AWS Transit Gatewayconnecter VPCs des réseaux sur site via un hub central. Chacune de ces options peut permettre à Redshift de communiquer avec un cluster Amazon MSK ou avec Amazon MSK Serverless dans un autre VPC.
Utilisation et activation de l'actualisation automatique	Les requêtes d'actualisation automatique pour une ou plusieurs vues matérialisées sont traitées comme toute autre charge de travail utilisateur. L'actualisation automatique charge les données du flux à mesure qu'elles arrivent. L'actualisation automatique peut être activée explicitement pour une vue matérialisée créée pour l'ingestion en streaming . Pour ce faire, spécifiez `AUTO REFRESH` dans la définition de la vue matérialisée. Par défaut, l'actualisation est manuelle. Pour spécifier l'actualisation automatique pour une vue matérialisée existante à des fins d'ingestion en streaming, vous pouvez exécuter `ALTER MATERIALIZED VIEW` pour l'activer. Pour en savoir plus, consultez CREATE MATERIALIZED VIEW ou ALTER MATERIALIZED VIEW.
Ingestion du streaming et Amazon Redshift Serverless	Les instructions d'installation et de configuration qui s'appliquent à l'ingestion du streaming par Amazon Redshift sur un cluster provisionné s'appliquent également à l'ingestion du streaming sur Amazon Redshift Serverless. Il est important de spécifier le niveau nécessaire pour RPUs prendre en charge l'ingestion du streaming avec l'actualisation automatique et d'autres charges de travail. Pour obtenir plus d'informations, consultez Facturation d'Amazon Redshift sans serveur.

Fonction ou comportement	Description
Nœuds Amazon Redshift situés dans une zone de disponibilité différente de celle du cluster Amazon MSK	Lorsque vous configurez l'ingestion du streaming, Amazon Redshift tente de se connecter à un cluster Amazon MSK situé dans la même zone de disponibilité, si la reconnaissance du rack est activée pour Amazon MSK. Si tous vos nœuds se trouvent dans des zones de disponibilité différentes de celles de votre cluster Amazon Redshift, vous pouvez avoir à payer des frais de transfert de données entre zones de disponibilité. Pour éviter cela, conservez au moins un nœud de cluster de courtiers Amazon MSK dans la même zone que votre cluster ou groupe de travail provisionné par Redshift.
Actualiser l'emplacement de départ	Après avoir créé une vue matérialisée, son actualisation initiale commence à partir `TRIM_HORIZON` d'un flux Kinesis ou à partir du décalage 0 d'une rubrique Amazon MSK.
Formats de données	Les formats de données pris en charge sont limités à ceux à partir desquels la conversion est possible`VARBYTE`. Pour plus d'informations, consultez Type VARBYTE et Opérateurs VARBYTE.
Ajouter des enregistrements à une table	Vous pouvez exécuter `ALTER TABLE APPEND` pour ajouter des lignes à une table cible à partir d'une vue matérialisée source existante. Cela ne fonctionne que si la vue matérialisée est configurée pour l'ingestion en streaming. Pour plus d'informations, consultez ALTER TABLE APPEND.

Fonction ou comportement	Description
Exécution de TRUNCATE ou DELETE	Vous pouvez supprimer des enregistrements d'une vue matérialisée utilisée pour l'ingestion de flux en continu, en procédant comme suit : • `TRUNCATE`— Cela supprime toutes les lignes d'une vue matérialisée configurée pour l'ingestion du streaming. Elle n’effectue pas d’analyse de la table. Pour plus d’informations, consultez TRUNCATE. • `DELETE`— Cela supprime toutes les lignes d'une vue matérialisée configurée pour l'ingestion du streaming. Pour plus d’informations, consultez DELETE.

Fonction ou comportement	Description
Identifiants non minuscules	Lorsque vous créez des vues matérialisées en streaming sur des sujets Amazon Managed Streaming for Apache Kafka ou Kinesis Data Streams contenant des identifiants autres que des minuscules, les actualisations automatiques peuvent échouer. Pour résoudre ce problème, procédez de l'une des manières suivantes : • Utilisez des actualisations manuelles avec le `enable_case_sensitive_identifier` paramètre défini sur `true` • Activez les actualisations automatiques en les définissant `enable_case_sensitive_identifier` au `true` niveau de la base de données ou du cluster **Note** Le réglage `enable_case_sensitive_identifier` au niveau de l'utilisateur n'est pas suffisant pour les actualisations automatiques, mais fonctionnera pour les actualisations manuelles. Pour plus d'informations sur les identificateurs distinguant les majuscules et minuscules, consultez. enable_case_sensitive_identifier

Mise en route de l'ingestion en streaming à partir d'Amazon Kinesis Data Streams

Cette rubrique explique comment utiliser les données de streaming issues de Kinesis Data Streams à l'aide d'une vue matérialisée.

La configuration de l'ingestion en streaming Amazon Redshift implique la création d'un schéma externe qui est mappé à la source de données de streaming, et la création d'une vue matérialisée faisant référence au schéma externe. L'ingestion en streaming Amazon Redshift prend en charge

Kinesis Data Streams en tant que source. Par conséquent, vous devez disposer d'une source Kinesis Data Streams disponible avant de configurer l'ingestion en streaming. Si vous n'avez pas de source, suivez les instructions décrites dans Premiers pas avec Amazon Kinesis Data Streams dans la documentation Kinesis, ou créez-en une sur la console en suivant les instructions fournies dans Création d'un flux via la Console de gestion AWS.

L'ingestion en streaming Amazon Redshift utilise une vue matérialisée, qui est mise à jour directement à partir du flux lors de l'exécution de `REFRESH`. La vue matérialisée est mappée à la source de données du flux. Vous pouvez effectuer des filtrages et agrégations sur les données de flux dans le cadre de la définition de la vue matérialisée. Votre vue matérialisée d'ingestion en streaming (la vue matérialisée de base) ne peut référencer qu'un seul flux. Toutefois, vous pouvez créer des vues matérialisées supplémentaires qui se joignent à la vue matérialisée de base et à d'autres vues ou tables matérialisées.

> **Note**
>
> Ingestion en streaming et Amazon Redshift Serverless – Les étapes de configuration décrites dans cette rubrique s'appliquent aux clusters provisionnés Amazon Redshift et à Amazon Redshift Serverless. Pour de plus amples informations, veuillez consulter Comportement d'ingestion du streaming et types de données.

En partant du principe que vous avez un flux Kinesis Data Streams à disposition, la première étape consiste à définir un schéma dans Amazon Redshift avec `CREATE EXTERNAL SCHEMA` et de référencer une ressource Kinesis Data Streams. Ensuite, pour accéder aux données du flux, définissez le `STREAM` dans une vue matérialisée. Vous pouvez stocker des registres de flux au format `SUPER` semi-structuré, ou définir un schéma qui entraîne la conversion des données en types de données Redshift. Lorsque vous interrogez la vue matérialisée, les enregistrements renvoyés sont une point-in-time vue du flux.

1. Créez un rôle IAM avec une politique d'approbation permettant à votre cluster Amazon Redshift ou à votre groupe de travail Amazon Redshift sans serveur d'endosser le rôle. Pour plus d'informations sur la façon de configurer la politique de confiance pour le rôle IAM, consultez Autoriser Amazon Redshift à accéder à AWS d'autres services en votre nom. Une fois créé, le rôle doit posséder la politique IAM suivante, qui offre l'autorisation de communiquer avec le flux de données Amazon Kinesis.

   Politique IAM pour un flux non chiffré provenant de Kinesis Data Streams

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadStream",
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStreamSummary",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards",
                "kinesis:DescribeStream"
            ],
            "Resource": "arn:aws:kinesis:*:0123456789:stream/*"
        },
        {
            "Sid": "ListStream",
            "Effect": "Allow",
            "Action": "kinesis:ListStreams",
            "Resource": "*"
        }
    ]
}
```

Politique IAM pour un flux chiffré provenant de Kinesis Data Streams

```
{
 "Version": "2012-10-17",
 "Statement": [{
   "Sid": "ReadStream",
   "Effect": "Allow",
   "Action": [
    "kinesis:DescribeStreamSummary",
    "kinesis:GetShardIterator",
    "kinesis:GetRecords",
    "kinesis:ListShards",
    "kinesis:DescribeStream"
   ],
   "Resource": "arn:aws:kinesis:*:0123456789:stream/*"
  },
  {
   "Sid": "DecryptStream",
```

```
   "Effect": "Allow",
   "Action": [
    "kms:Decrypt"
   ],
   "Resource": "arn:aws:kms:us-
east-1:0123456789:key/1234abcd-12ab-34cd-56ef-1234567890ab"
  },
        {
   "Sid": "ListStream",
   "Effect": "Allow",
   "Action": "kinesis:ListStreams",
   "Resource": "*"
  }
 ]
}
```

2. Vérifiez votre VPC et que votre cluster Amazon Redshift ou Amazon Redshift sans serveur dispose d'une route permettant d'accéder aux points de terminaison Kinesis Data Streams via Internet à l'aide d'une passerelle NAT ou d'une passerelle Internet. Si vous souhaitez que le trafic entre Redshift et Kinesis Data Streams reste sur le AWS réseau, pensez à utiliser un point de terminaison VPC Kinesis Interface. Pour plus d'informations, consultez Using Amazon Kinesis Data Streams Kinesis Data Streams with Interface VPC Endpoints (Utilisation d'Amazon Kinesis Data Streams Kinesis Data Streams avec des points de terminaison d'un VPC d'interface).
3. Dans Amazon Redshift, créez un schéma externe afin de mapper les données de Kinesis à un schéma.

```
CREATE EXTERNAL SCHEMA kds
FROM KINESIS
IAM_ROLE { default | 'iam-role-arn' };
```

L'ingestion en streaming pour Kinesis Data Streams ne nécessite aucun type d'authentification. Elle utilise le rôle IAM défini dans l'instruction `CREATE EXTERNAL SCHEMA` pour effectuer des demandes Kinesis Data Streams.

Facultatif : utilisez le mot clé REGION pour spécifier la région dans laquelle réside le flux Amazon Kinesis Data Streams ou le flux Amazon MSK.

```
CREATE EXTERNAL SCHEMA kds
FROM KINESIS
REGION 'us-west-2'
```

```
IAM_ROLE { default | 'iam-role-arn' };
```

Dans cet exemple, la région indique l'emplacement du flux source. IAM_ROLE est un exemple.

4. Créez une vue matérialisée pour consommer les données du flux. Avec une instruction comme celle-ci, si un enregistrement ne peut pas être analysé, cela provoque une erreur. Utilisez une commande comme celle-ci si vous ne voulez pas que les enregistrements d'erreur soient ignorés.

```
CREATE MATERIALIZED VIEW my_view AUTO REFRESH YES AS
SELECT *
FROM kds.my_stream_name;
```

Les noms de flux Kinesis sont sensibles à la casse et peuvent contenir des lettres majuscules et minuscules. Pour effectuer une ingestion à partir de flux dont les noms sont en majuscules, vous pouvez définir la configuration `enable_case_sensitive_identifier` `true` au niveau de la base de données. Pour plus d'informations, consultez Noms et identificateurs et enable_case_sensitive_identifier.

Pour activer l'actualisation automatique, utilisez `AUTO REFRESH YES`. L'actualisation manuelle est le comportement par défaut. Notez que lorsque vous utilisez CAN_JSON_PARSE, il est possible que les enregistrements qui ne peuvent pas être analysés soient ignorés.

Les colonnes de métadonnées incluent ce qui suit :

Colonne de métadonnées	Type de données	Description
approximate_arrival_timestamp	horodatage sans fuseau horaire	L'heure approximative à laquelle l'enregistrement a été inséré dans le flux Kinesis
partition_key	varchar(256)	La clé utilisée par Kinesis pour attribuer l'enregistrement à une partition
shard_id	char(20)	L'identifiant unique de la partition dans le flux à partir

Colonne de métadonnées	Type de données	Description
		duquel l’enregistrement a été extrait
sequence_number	varchar(128)	L’identifiant unique de l’enregistrement à partir de la partition Kinesis
refresh_time	horodatage sans fuseau horaire	L’heure de début de l’actualisation
kinesis_data	varbyte	L’enregistrement à partir du flux Kinesis

Il est important de noter que si la définition de votre vue matérialisée intègre une logique métier, les erreurs de logique métier peuvent bloquer l'ingestion du streaming dans certains cas. Cela peut vous obliger à supprimer et à recréer la vue matérialisée. Pour éviter cela, nous vous recommandons de garder votre logique aussi simple que possible et d'effectuer la plupart de vos vérifications de logique métier sur les données après leur ingestion.

5. Actualisez la vue, ce qui invoque Redshift pour qu’il lise à partir du flux et charge les données dans la vue matérialisée.

```
REFRESH MATERIALIZED VIEW my_view;
```

6. Interrogez les données dans la vue matérialisée.

```
select * from my_view;
```

Commencer à ingérer du streaming à partir de sources Apache Kafka

Cette rubrique explique comment utiliser des données de streaming depuis Amazon MSK, Apache Kafka ou Confluent Cloud à l'aide d'une vue matérialisée.

L’objectif de l’ingestion en streaming Amazon Redshift est de simplifier le processus afin d’ingérer directement des données de flux depuis un service de streaming vers Amazon Redshift ou Amazon Redshift sans serveur. Cela fonctionne avec Amazon MSK Provisioned et Amazon MSK Serverless,

avec l'open source Apache Kafka et avec Confluent Cloud. L'ingestion du streaming par Amazon Redshift élimine le besoin de créer un sujet Apache Kafka dans Amazon S3 avant d'ingérer les données du flux dans Redshift.

Sur le plan technique, l'ingestion du streaming permet une ingestion rapide et à faible latence des données de flux ou de sujets dans une vue matérialisée Amazon Redshift. Après la configuration, l'actualisation de la vue matérialisée vous permet d'intégrer de gros volumes de données.

Vous devez disposer d'une source Apache Kafka avant de configurer l'ingestion du streaming par Amazon Redshift. Si vous n'avez pas de source, créez-en une en suivant les instructions suivantes :

- Amazon MSK — Commencer à utiliser Amazon MSK
- Apache Kafka — Démarrage rapide d'Apache Kafka
- Confluent Cloud — Démarrage rapide pour Confluent Cloud

Configuration de l'ingestion du streaming depuis Kafka

Utilisez les procédures suivantes pour configurer l'ingestion de flux vers Amazon Redshift à partir d'Amazon MSK ou de sources Apache Kafka non AWS gérées (Apache Kafka et Confluent Cloud).

Rubriques

- Configurer l'authentification
- Configuration de votre VPC
- Création d'une vue matérialisée

Configurer l'authentification

Cette section décrit la configuration de l'authentification pour permettre à votre application Amazon Redshift d'accéder à une source Amazon MSK.

Après avoir créé le rôle de votre application, associez l'une des politiques suivantes pour autoriser l'accès à votre cluster Amazon MSK, Apache Kafka ou Confluent Cloud. Pour l'authentification mTLS, vous pouvez stocker les certificats utilisés par Amazon Redshift dans ACM ou Secrets Manager. Vous devez donc choisir la politique correspondant à l'endroit où le certificat est stocké.

Le contrôle d'accès IAM pour Amazon Redshift est disponible pour les clusters MSK dotés de la version 2.7.1 ou supérieure de Kafka.

AUTHENTIFICATION IAM (Amazon MSK uniquement) :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "MSKIAMpolicy",
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:ReadData",
                "kafka-cluster:DescribeTopic",
                "kafka-cluster:Connect"
            ],
            "Resource": [
                "arn:aws:kafka:*:0123456789:cluster/MyTestCluster/*",
                "arn:aws:kafka:*:0123456789:topic/MyTestCluster/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "kafka-cluster:AlterGroup",
                "kafka-cluster:DescribeGroup"
            ],
            "Resource": [
                "arn:aws:kafka:*:0123456789:group/MyTestCluster/*"
            ]
        }
    ]
}
```

AUTHENTIFICATION MTLS : à l'aide d'un certificat stocké dans AWS Certificate Manager

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "MSKmTLSACMpolicy",
            "Effect": "Allow",
            "Action": [
                "acm:ExportCertificate"
            ],
            "Resource": [
                "arn:aws:acm:us-east-1:444455556666:certificate/certificate_ID"
            ]
```

```
        }
    ]
}
```

AUTHENTIFICATION MTLS : à l'aide d'un certificat stocké dans AWS Secrets Manager

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "MSKmTLSSecretsManagerpolicy",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:secretsmanager:us-east-1:444455556666:secret:secret_ID"
            ]
        }
    ]
}
```

Amazon MSK

Si vous utilisez AUTHENTICATION NONE pour vous connecter à une source Amazon MSK, aucun rôle IAM n'est requis. Toutefois, si vous utilisez AUTHENTICATION IAM ou MTLS pour vous authentifier auprès de votre cluster Amazon MSK, votre cluster Amazon Redshift ou votre espace de noms Amazon Redshift Serverless doit être associé à un rôle IAM avec les autorisations appropriées. Créez un rôle IAM avec une politique de confiance qui permet à votre cluster Amazon Redshift ou à votre espace de noms Amazon Redshift Serverless d'assumer ce rôle. Après avoir créé le rôle, ajoutez l'une des autorisations suivantes pour prendre en charge IAM ou MTLS. Pour l'authentification mTLS, les certificats utilisés par Amazon Redshift peuvent être stockés AWS Certificate Manager dans AWS Secrets Manager ou. Vous devez donc choisir la politique correspondant à l'endroit où le certificat est stocké. Associez le rôle à votre cluster provisionné Amazon Redshift ou à votre espace de noms Redshift Serverless. Pour plus d'informations sur la façon de configurer la politique de confiance pour le rôle IAM, consultez Autoriser Amazon Redshift à accéder à AWS d'autres services en votre nom.

Le tableau suivant montre les options de configuration complémentaires à définir pour l’ingestion en streaming depuis Amazon MSK :

Configuration Amazon Redshift	Configuration d'Amazon MSK	Port à ouvrir entre Redshift et Amazon MSK
AUTHENTICATION NONE	Transport TLS désactivé	9092
AUTHENTICATION NONE	Transport TLS activé	9094
AUTHENTICATION IAM	IAM	9098/9198
AUTHENTIFICATION MTLS	Transport TLS activé	9094

L'authentification Amazon Redshift est définie dans l'instruction CREATE EXTERNAL SCHEMA.

> **Note**
>
> Si l'authentification mTLS (Mutual Transport Layer Security) est activée sur le cluster Amazon MSK, la configuration d'Amazon Redshift pour utiliser AUTHENTICATION NONE lui indique d'utiliser le port 9094 pour un accès non authentifié. Toutefois, cela échouera, car le port est utilisé par l'authentification mTLS. Pour cette raison, nous vous recommandons de passer à AUTHENTIFICATION mtls lorsque vous utilisez mTLS.

Apache Kafka or Confluent Cloud

Pour Apache Kafka et Confluent Cloud, Amazon Redshift prend en charge les protocoles de connexion suivants :

- Vous pouvez utiliser le protocole MTL ou le texte brut avec le transport TLS pour l'authentification lorsque vous vous connectez à Apache Kafka.
- Vous ne pouvez utiliser mTLS que pour l'authentification lorsque vous vous connectez à Confluent Cloud.

Amazon Redshift prend en charge les protocoles de chiffrement suivants pour la connexion à Apache Kafka ou à Confluent Cloud :

Méthodes d'authentification prises en charge pour Apache Kafka et Confluent Cloud

Amazon Redshift	Protocole de sécurité Kafka	Support d'Apache Kafka	Assistance Confluent Cloud
AUTHENTICATION NONE	TEXTE EN CLAIR	Non	Non
AUTHENTICATION NONE	SSL	Oui	Non
AUTHENTICATION IAM	SASL_SSL	Non	Non
AUTHENTIF ICATION MTLS	SSL	Oui (avec certificat)	Oui (avec certificat)

Notez qu'Amazon Redshift ne prend pas en charge SASL/SCRAM le protocole SASL/PLAINTEXT.

Configuration de votre VPC

Après avoir créé vos ressources d'authentification, vérifiez votre VPC et vérifiez que votre cluster Amazon Redshift ou votre groupe de travail Amazon Redshift Serverless dispose d'un itinéraire pour accéder à votre source Apache Kafka.

> **Note**
>
> Pour Amazon MSK, les règles du groupe de sécurité entrant pour votre cluster Amazon MSK doivent autoriser le groupe de sécurité de votre cluster Amazon Redshift ou de votre groupe de travail Redshift Serverless. Les ports que vous spécifiez dépendent des méthodes d'authentification configurées sur votre cluster Amazon MSK. Pour plus d'informations, consultez les sections Informations sur les ports et Accès depuis le VPC AWS mais depuis l'extérieur.

Activez ensuite le routage VPC amélioré sur votre cluster Amazon Redshift ou votre groupe de travail Amazon Redshift Serverless. Pour plus d'informations, consultez Activation du routage VPC amélioré.

Création d'une vue matérialisée

Dans cette section, vous allez configurer la vue matérialisée qu'Amazon Redshift utilise pour accéder à vos données de streaming Apache Kafka.

En supposant qu'un cluster Apache Kafka soit disponible, la première étape consiste à définir un schéma dans Redshift `CREATE EXTERNAL SCHEMA` avec et à référencer le cluster comme source de données. Ensuite, pour accéder aux données dans la rubrique, définissez le `STREAM` dans une vue matérialisée. Vous pouvez stocker les enregistrements de votre rubrique en utilisant le type de données Amazon Redshift VARBYTE par défaut, ou définir un schéma qui convertit les données au format semi-structuré. `SUPER` Lorsque vous interrogez la vue matérialisée, les enregistrements renvoyés sont une point-in-time vue du sujet.

1. Dans Amazon Redshift, créez un schéma externe à mapper au cluster Apacke Kafka. La syntaxe est la suivante :

   ```
   CREATE EXTERNAL SCHEMA MySchema
   FROM KAFKA
   [ IAM_ROLE [ default | 'iam-role-arn' ] ]
   AUTHENTICATION [ none | iam | mtls ]
   {AUTHENTICATION_ARN 'acm-certificate-arn' |  SECRET_ARN 'asm-secret-arn'};
   ```

   Dans la `FROM` clause, `KAFKA` indique que le schéma mappe les données d'une source Apache Kafka.

   `AUTHENTICATIO`Nindique le type d'authentification pour l'ingestion du streaming. Trois types sont disponibles :

   - none — Spécifie qu'aucune authentification n'est requise. Cela correspond à un accès non authentifié sur MSK. Cela correspond à l'authentification SSL dans Apache Kafka. Cette méthode d'authentification n'est pas prise en charge pour Confluent Cloud.
   - iam – Indique une authentification IAM. Vous ne pouvez utiliser l'authentification IAM qu'avec Amazon MSK. Lorsque vous choisissez cette option, vérifiez que le rôle IAM dispose des autorisations nécessaires à l’authentification IAM. Pour plus d'informations sur la configuration des politiques IAM requises, consultezConfiguration de l'ingestion du streaming depuis Kafka.
   - mtls — Spécifie que la sécurité mutuelle de la couche de transport assure une communication sécurisée en facilitant l'authentification entre un client et un serveur. Dans ce cas, le client est Redshift et le serveur est Apache Kafka. Pour plus d'informations sur la configuration de

l'ingestion du streaming avec les mTLS, consultezAuthentification avec MTL pour l'ingestion de flux Redshift à partir de sources Apache Kafka.

Notez que l'authentification Amazon MSK avec un nom d'utilisateur et un mot de passe n'est pas prise en charge pour l'ingestion du streaming.

Le `AUTHENTICATION_ARN` paramètre spécifie l'ARN du certificat mTLS (ACM Mutual Transport Layer Security) que vous utilisez pour établir une connexion cryptée.

Le `SECRET_ARN` paramètre spécifie l'ARN du AWS Secrets Manager secret contenant le certificat à utiliser par Amazon Redshift pour les MTL.

Les exemples suivants montrent comment définir l'URI du courtier pour le cluster Amazon MSK lorsque vous créez le schéma externe :

Utilisation de l'authentification IAM :

```
CREATE EXTERNAL SCHEMA my_schema
FROM KAFKA
IAM_ROLE 'arn:aws:iam::012345678901:role/my_role'
AUTHENTICATION IAM
URI 'b-1.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9098,b-2.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9098'
```

Sans authentification :

```
CREATE EXTERNAL SCHEMA my_schema
FROM KAFKA
AUTHENTICATION none
URI 'b-1.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9092,b-2.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9092'
```

À l'aide de MTL :

```
CREATE EXTERNAL SCHEMA my_schema
FROM KAFKA
IAM_ROLE 'arn:aws:iam::012345678901:role/my_role'
AUTHENTICATION MTLS
```

```
URI 'b-1.myTestCluster.123z8u.c2.kafka.us-west-1.amazonaws.com:9094,b-
 2.myTestCluster.123z8u.c2.kafka.us-west-1.amazonaws.com:9094'
{AUTHENTICATION_ARN 'acm-certificate-arn' |  SECRET_ARN 'asm-secret-arn'}
```

Pour plus d'informations sur la création d'un schéma externe, voir CREATE EXTERNAL SCHEMA.

2. Créez une vue matérialisée pour consommer les données de la rubrique. Utilisez une commande SQL telle que l'exemple suivant.

```
CREATE MATERIALIZED VIEW MyView AUTO REFRESH YES AS
SELECT *
FROM MySchema."mytopic";
```

Les noms de rubrique Kafka sont sensibles à la casse et peuvent contenir des lettres majuscules et minuscules. Pour effectuer une ingestion à partir de rubriques dont les noms sont en majuscules, vous pouvez définir la configuration `enable_case_sensitive_identifier true` au niveau de la session ou de la base de données. Pour plus d'informations, consultez Noms et identificateurs et enable_case_sensitive_identifier.

Pour activer l'actualisation automatique, utilisez `AUTO REFRESH YES`. L'actualisation manuelle est le comportement par défaut.

3. Les colonnes de métadonnées incluent ce qui suit :

Colonne de métadonnées	Type de données	Description
kafka_partition	bigint	ID de partition de l'enregis trement issu de la rubrique Kafka
kafka_offset	bigint	Décalage de l'enregistrement dans la rubrique Kafka pour une partition donnée
kafka_timestamp_type	char(1)	Type d'horodatage utilisé dans l'enregistrement Kafka :

Colonne de métadonnées	Type de données	Description
		• C – Enregistrer l’heure de création (CREATE_TIME) côté client • L – Enregistrer l’heure d’ajout (LOG_APPE ND_TIME) côté serveur Kafka • U – Enregistrer l’heure de création indisponible (NO_TIMESTAMP_TYPE)
kafka_timestamp	horodatage sans fuseau horaire	Valeur timestamp de l’enregistrement
kafka_key	varbyte	La clé de l’enregistrement de Kafka
kafka_value	varbyte	L’enregistrement reçu de Kafka
kafka_headers	super	L’en-tête de l’enregistrement reçu de Kafka
refresh_time	horodatage sans fuseau horaire	L’heure de début de l’actualisation

Il est important de noter que si la définition de votre vue matérialisée comporte une logique métier qui entraîne des erreurs de logique métier, cela peut entraîner des échecs d'ingestion lors de l'ingestion du streaming dans certains cas. Cela peut vous obliger à supprimer et à recréer la vue matérialisée. Pour éviter cela, nous vous recommandons de conserver une logique métier simple et d'exécuter une logique supplémentaire sur les données après les avoir ingérées.

4. Actualisez la vue, ce qui appelle Amazon Redshift à lire à partir de la rubrique et à charger des données dans la vue matérialisée.

```
REFRESH MATERIALIZED VIEW MyView;
```

5. Interrogez les données dans la vue matérialisée.

```
select * from MyView;
```

La vue matérialisée est mise à jour directement à partir de la rubrique lorsque REFRESH est exécuté. Vous créez une vue matérialisée qui se mappe à la source de données de la rubrique Kafka. Vous pouvez effectuer des filtrages et des agrégations sur les données dans le cadre de la définition de la vue matérialisée. Votre vue matérialisée de l'ingestion en streaming (la vue matérialisée de base) ne peut référencer qu'une seule rubrique Kafka. Toutefois, vous pouvez créer des vues matérialisées supplémentaires qui se joignent à la vue matérialisée de base et à d'autres vues ou tables matérialisées.

Pour plus d'informations sur les limites relatives à l'ingestion en streaming, consultez Comportement d'ingestion du streaming et types de données.

Authentification avec MTL pour l'ingestion de flux Redshift à partir de sources Apache Kafka

La sécurité mutuelle de la couche de transport (mTLS) permet à un serveur d'authentifier un client auquel il envoie des informations et au client d'authentifier le serveur. L'avantage de l'utilisation des MTL est de fournir une authentification fiable pour une variété de cas d'utilisation dans plusieurs applications sectorielles. Il s'agit notamment de cas d'utilisation dans les secteurs de la finance, du commerce de détail, du gouvernement et de la santé. En cas d'ingestion du streaming vers Redshift, l'authentification a lieu entre un serveur, qui peut être Amazon MSK, Apache Kafka ou Confluent Cloud, et un cluster provisionné par Amazon Redshift ou un groupe de travail Amazon Redshift Serverless.

Cette rubrique fournit des procédures et des exemples de commandes SQL qui montrent comment créer un schéma externe qui utilise mTLS pour s'authentifier entre le client Redshift et n'importe quel serveur Apache Kafka. Les étapes décrites dans cette rubrique complètent l'ensemble des étapes de configuration de l'ingestion du streaming à partir de sources Apache Kafka. Pour de plus amples informations, veuillez consulter Commencer à ingérer du streaming à partir de sources Apache Kafka.

Conditions préalables à l'utilisation des MTL pour l'ingestion du streaming

Cette section fournit les étapes préalables à l'utilisation des MTL pour l'ingestion du streaming avec AWS Certificate Manager ou AWS Secrets Manager.

À titre d'étape préliminaire, vous devez disposer ou créer une autorité de certification privée (PCA), que vous pouvez utiliser pour émettre des certificats qui, entre autres fonctions, permettent une communication sécurisée via des canaux de communication sécurisés. AWS Private Certificate Authority (Private CA) est un service disponible qui exécute cette fonction. Pour plus d'informations, consultez la section Création d'une autorité de certification privée dans le guide de AWS Private Certificate Authority l'utilisateur. Après avoir créé l'autorité de certification privée, exportez le certificat de l'autorité de certification racine et enregistrez-le dans un fichier portant l'extension .pem.

Pour créer un cluster utilisant le certificat CA, procédez comme suit :

Amazon MSK

1. Créez un cluster Amazon MSK qui prend en charge l'authentification client MTLS. Pour plus d'informations sur la configuration d'un cluster Amazon MSK, consultez la section Authentification mutuelle des clients TLS pour Amazon MSK dans le manuel Amazon Managed Streaming for Apache Kafka Developer Guide.
2. Modifiez les paramètres de sécurité du cluster Amazon MSK, en activant l'authentification client TLS à l'aide de AWS Certificate Manager (ACM) et en sélectionnant le AWS Private CA (PCA) que vous avez créé précédemment. Pour plus d'informations, consultez la section Mise à jour des paramètres de sécurité d'un cluster dans le manuel Amazon Managed Streaming for Apache Kafka Developer Guide.

Confluent Cloud

1. Créez un cluster Confluent Cloud dédié, de préférence Région AWS identique à votre cluster Amazon Redshift. Pour plus d'informations sur la création d'un cluster Confluent Cloud, voir Création d'un cluster Kafka dans Confluent Cloud.
2. Téléchargez le fichier pem du certificat AWS Private CA racine CA exporté que vous avez créé précédemment. Pour plus d'informations, consultez Gérer l'autorité de certification pour l'authentification mTLS pour Confluent Cloud. Confluent Cloud utilise ce certificat pour vérifier le certificat client Amazon Redshift.

Utilisation de MTL pour l'ingestion du streaming avec AWS Certificate Manager

La procédure suivante explique comment configurer les MTL pour l'ingestion du streaming Redshift en AWS Certificate Manager utilisant (ACM) pour le stockage et la gestion des certificats :

1. Demandez un certificat privé via ACM. Dans ce cas, sélectionnez le PCA que vous avez créé dans la section Prérequis comme autorité de certification. ACM stocke le certificat signé et la clé privée jointe pour une communication sécurisée. Pour plus d'informations sur la gestion des certificats avec ACM, consultez la section Émission et gestion des certificats dans le guide de l'AWS Certificate Manager utilisateur.
2. Pour le rôle IAM que vous utilisez pour gérer votre cluster Redshift ou votre groupe de travail Amazon Redshift Serverless, attachez l'autorisation d'exporter le certificat, qui est acm :. ExportCertificate Pour plus d'informations sur la configuration des ressources IAM nécessaires à l'ingestion du streaming, consultezConfiguration de l'ingestion du streaming depuis Kafka. Spécifiez le même rôle IAM à l'étape suivante pour créer le schéma externe.

   > **Note**
   >
   > Demandes pour AWS Certificate Manager exiger une passerelle Internet (IGW) ou une passerelle NAT (NGW) dans votre VPC. Si votre VPC ne possède ni IGW ni NGW, procédez comme suit :
   >
   > - Utilisez Secrets Manager au lieu d'ACM pour stocker vos certificats.
   > - Attachez un point de terminaison VPC Secrets Manager à votre VPC.
   >
   > Pour plus d'informations sur l'utilisation de Secrets Manager avec MTL pour l'ingestion du streaming, voir Utilisation de MTL pour l'ingestion du streaming avec AWS Secrets Manager ci-dessous.

3. Obtenez l'URI du broker bootstrap pour le cluster Amazon MSK, Apache Kafka ou Confluent Cloud. Pour plus d'informations sur l'obtention de l'URI du courtier bootstrap pour Amazon MSK, consultez Getting the bootstrap brokers for an Amazon MSK cluster dans le manuel Amazon Managed Streaming for Apache Kafka Developer Guide.
4. Exécutez une commande SQL telle que l'exemple suivant pour créer un schéma externe qui mappe le cluster à un schéma externe Redshift, en utilisant. `mtls`

Amazon MSK

```
CREATE EXTERNAL SCHEMA my_schema
FROM KAFKA
IAM_ROLE 'arn:aws:iam::012345678901:role/my_role'
AUTHENTICATION mtls
URI 'b-1.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9094,b-2.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9094'
AUTHENTICATION_ARN 'arn:aws:acm:Region:444455556666:certificate/certificate_ID';
```

Apache Kafka or Confluent Cloud

```
CREATE EXTERNAL SCHEMA my_schema
FROM KAFKA
IAM_ROLE 'arn:aws:iam::012345678901:role/my_role'
AUTHENTICATION mtls
URI 'lkc-2v531.domz6wj0p.us-west-1.aws.confluent.cloud:9092'
AUTHENTICATION_ARN 'arn:aws:acm:region:444455556666:certificate/certificate_ID';
```

Paramètres importants :

- IAM_ROLE — Rôle IAM associé au cluster, pour l'ingestion du streaming.
- URI — L'URI du courtier bootstrap pour le cluster. Notez que pour Amazon MSK, le port 9094 est spécifié pour communiquer avec les courtiers pour le chiffrement TLS.
- AUTHENTICATION_ARN — L'ARN du certificat ACM. L'ARN est disponible dans la console ACM lorsque vous choisissez le certificat émis.

Après avoir effectué ces étapes de configuration, vous pouvez créer une vue matérialisée Redshift qui fait référence au schéma défini dans l'exemple, puis utiliser REFRESH MATERIALIZED VIEW pour diffuser des données. Pour de plus amples informations, veuillez consulter Commencer à ingérer du streaming à partir de sources Apache Kafka.

Utilisation de MTL pour l'ingestion du streaming avec AWS Secrets Manager

Vous pouvez configurer les MTL pour l'ingestion du streaming Redshift en les AWS Secrets Manager utilisant pour la gestion des certificats si vous ne souhaitez pas référencer le certificat dans. AWS

Certificate Manager Les étapes suivantes décrivent comment configurer les MTLs à l'aide de Secrets Manager.

1. Créez une demande de signature de certificat et une clé privée avec l'outil de votre choix. Vous pouvez ensuite utiliser la demande de signature pour générer un certificat signé, en utilisant la même autorité de certification AWS privée (PCA) que celle que vous avez utilisée pour générer le certificat pour le cluster. Pour plus d'informations sur l'émission d'un certificat, consultez IssueCertificatela référence de l'AWS Private Certificate Authority API.
2. Extrayez le certificat à l'aide de AWS Private Certificate Authority. Pour plus d'informations, consultez la section Récupérer un certificat privé dans le guide de AWS Private Certificate Authority l'utilisateur.
3. Stockez le certificat et la clé privée générés à l'étape précédente dans AWS Secrets Manager. Choisissez `Other type of secret` et utilisez le format de texte brut. Les paires clé-valeur doivent être au format`{"certificate":"<cert value>","privateKey":"<pkey value>"}`, comme dans l'exemple suivant. Pour plus d'informations sur la création et la gestion de secrets dans AWS Secrets Manager, voir Création et gestion de secrets AWS Secrets Manager dans le Guide de AWS Secrets Manager l'utilisateur.

   ```
   {"certificate":"-----BEGIN CERTIFICATE-----
   klhdslkfjahksgdfkgioeuyihbflahabhbdslv6akybeoiwv1hoaiusdhbahsbdi
   H4hAX8/eE96qCcjkpfT84EdvHzp6fC+/WwM0oXlwUEWlvfMCXNaG5D8SqRq3qA==
   -----END CERTIFICATE-----
   -----BEGIN CERTIFICATE-----
   klhdslkfjahksgdfkgioeuyihbflahabhbdslv6akybeoiwv1hoaiusdhbahsbdi
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   -----END CERTIFICATE-----",
   "privateKey":"-----BEGIN PRIVATE KEY-----
   klhdslkfjahksgdfkgioeuyihbflahabhbdslv6akybeoiwv1hoaiusdhbahsbdi
   7OD4m1dBEs3Fj++hDMH9rYRp99RqtCOf0EWOUe139KOilOsW+cyhAoc9Ci2+Jo/k
   17u2N1iGILMQEZuCRtnJOkFYkw==
   -----END PRIVATE KEY-----"}
   ```

4. Joignez la politique d'autorisation pour récupérer le secret au rôle IAM que vous utilisez pour gérer votre cluster Amazon Redshift ou votre groupe de travail Amazon Redshift Serverless. Cette autorisation est`secretsmanager:GetSecretValue`. Pour de plus amples informations, veuillez consulter Configurer l'authentification. Pour plus d'informations sur la gestion des politiques IAM, voir Modifier les politiques IAM. Spécifiez le même rôle IAM à l'étape suivante pour créer le schéma externe.

5. Dans Redshift, exécutez la commande SQL pour créer le schéma externe. Vous utilisez le type d'AUTHENTIFICATION`mtls`. Vous spécifiez également l'URI du cluster et l'ARN secret dans AWS Secrets Manager.

```
CREATE EXTERNAL SCHEMA my_schema
FROM KAFKA
IAM_ROLE 'arn:aws:iam::012345678901:role/my_role'
AUTHENTICATION mtls
URI 'b-1.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9094,b-2.myTestCluster.123z8u.c2.kafka.us-
west-1.amazonaws.com:9094'
SECRET_ARN 'arn:aws:secretsmanager:us-east-1:012345678910:secret:myMTLSSecret';
```

Paramètres importants :

- IAM_ROLE — Rôle IAM associé au cluster, pour l'ingestion du streaming.
- URI — L'URI du courtier bootstrap pour le cluster. Notez que pour Amazon MSK, le port 9094 est spécifié pour communiquer avec les courtiers pour le chiffrement TLS.
- SECRET_ARN — L'ARN du secret provenant de Secrets Manager, contenant le certificat à utiliser pour les MTL.

Activation de l'authentification mTLS pour un schéma externe existant

Si vous utilisez un schéma externe pour l'ingestion du streaming et que vous souhaitez implémenter le protocole TLS mutuel pour l'authentification, vous pouvez exécuter une commande telle que la suivante, qui spécifie l'authentification mTLS et l'ARN du certificat ACM dans ACM.

```
ALTER EXTERNAL SCHEMA schema_name
AUTHENTICATION mtls
AUTHENTICATION_ARN 'arn:aws:acm:Region:444455556666:certificate/certificate_ID';
```

Vous pouvez également spécifier l'authentification mTLS, en référence à l'ARN secret dans AWS Secrets Manager.

```
ALTER EXTERNAL SCHEMA schema_name
AUTHENTICATION mtls
SECRET_ARN 'arn:aws:secretsmanager:us-east-1:012345678910:secret:myMTLSSecret';
```

Pour plus d'informations sur la commande ALTER EXTERNAL SCHEMA, consultezMODIFIER LE SCHÉMA EXTERNE.

Ingestion de données de streaming à l'aide de Kinesis

Cette procédure montre comment ingérer des données à partir d'un flux Kinesis nommé ev_station_data, qui contient les données de consommation de différentes bornes de recharge pour véhicules électriques, au format JSON. Le schéma est bien défini. L'exemple montre comment stocker les données sous forme de JSON brut et comment convertir les données JSON en types de données Amazon Redshift lorsqu'elles sont ingérées.

Configuration du producteur

1. À l'aide d'Amazon Kinesis Data Streams, suivez les étapes pour créer un flux nommé `ev_station_data`. Sélectionnez On-demand (À la demande) pour le Capacity mode (Mode de capacité). Pour plus d'informations, consultez la section Création d'un flux via la console AWS de gestion.
2. Amazon Kinesis Data Generator peut vous aider à générer des données de test à utiliser avec votre flux. Suivez les étapes détaillées dans l'outil pour commencer, et utilisez le modèle de données suivant afin de générer vos données :

```
{

   "_id" : "{{random.uuid}}",
   "clusterID": "{{random.number(
        {   "min":1,
            "max":50
        }
    )}}",
    "connectionTime": "{{date.now("YYYY-MM-DD HH:mm:ss")}}",
    "kWhDelivered": "{{commerce.price}}",
    "stationID": "{{random.number(
        {   "min":1,
            "max":467
        }
    )}}",
      "spaceID": "{{random.word}}-{{random.number(
        {   "min":1,
            "max":20
        }
    )}}",
```

```
   "timezone": "America/Los_Angeles",
   "userID": "{{random.number(
        {   "min":1000,
            "max":500000
        }
    )}}"
}
```

Chaque objet JSON des données du flux comprend les propriétés suivantes :

```
{
    "_id": "12084f2f-fc41-41fb-a218-8cc1ac6146eb",
    "clusterID": "49",
    "connectionTime": "2022-01-31 13:17:15",
    "kWhDelivered": "74.00",
    "stationID": "421",
    "spaceID": "technologies-2",
    "timezone": "America/Los_Angeles",
    "userID": "482329"
}
```

Configuration Amazon Redshift

Ces étapes vous montrent comment configurer la vue matérialisée afin d'ingérer des données.

1. Créez un schéma externe afin de mapper les données de Kinesis à un objet Redshift.

   ```
   CREATE EXTERNAL SCHEMA evdata FROM KINESIS
   IAM_ROLE 'arn:aws:iam::0123456789:role/redshift-streaming-role';
   ```

   Pour plus d'informations sur la configuration du rôle IAM, consultez Mise en route de l'ingestion en streaming à partir d'Amazon Kinesis Data Streams.

2. Créez une vue matérialisée pour consommer les données du flux. L'exemple suivant montre comment définir une vue matérialisée pour ingérer les données au format JSON à partir d'un flux Kinesis.

   Tout d'abord, stockez les registres de flux au format semi-structuré SUPER. Dans cet exemple, la source JSON est stockée dans Redshift sans conversion en types Redshift.

```
CREATE MATERIALIZED VIEW ev_station_data AS
    SELECT approximate_arrival_timestamp,
    partition_key,
    shard_id,
    sequence_number,
    case when can_json_parse(kinesis_data) then json_parse(kinesis_data) else null
 end as payload,
    case when not can_json_parse(kinesis_data) then kinesis_data else null end as
 failed_payload
    FROM evdata."ev_station_data" ;
```

Interrogation du flux

1. Activez les attributs SUPER distinguant majuscules et minuscules à l'aide de la commande ci-dessous. Amazon Redshift ne distingue pas les majuscules et minuscules par défaut. Pour accéder aux attributs SUPER qui distinguent les majuscules et minuscules, vous devez activer cette fonctionnalité.

   ```
   SET enable_case_sensitive_super_attribute to TRUE;
   ```

2. Actualisez la vue matérialisée à l'aide de la commande suivante afin d'extraire les données du flux.

   ```
   REFRESH MATERIALIZED VIEW ev_station_data;
   ```

3. Interrogez la vue matérialisée actualisée pour générer les statistiques d’utilisation.

   ```
   SELECT e.payload.connectionTime::date as connectiontime
   ,SUM(e.payload.kWhDelivered::decimal(10,2)) AS Energy_Consumed
   ,count(distinct e.payload.userID) AS #Users
   from ev_station_data as e
   group by connectiontime
   order by 1 desc;
   ```

4. Affichez les résultats.

   ```
   connectiontime        energy_consumed    #users
   2022-02-08                 4139          10
   2022-02-09                 5571          10
   2022-02-10                 8697          20
   ```

```
2022-02-11          4408          10
2022-02-12          4257          10
2022-02-23          6861          10
```

AWS Glue Data Catalog vues

Cette rubrique décrit comment créer des vues dans le AWS Glue Data Catalog. Vous pouvez utiliser les vues du catalogue de données pour accéder aux données de différentes sources de données en utilisant le même schéma.

En créant des vues dans le catalogue de données, vous pouvez créer un schéma de vue et un objet de métadonnées communs uniques à utiliser sur des moteurs tels qu'Amazon Athena et Amazon EMR Spark. Cela vous permet d'utiliser les mêmes vues sur l'ensemble de vos lacs de données et de vos entrepôts des données pour répondre à vos cas d'utilisation. Les vues du catalogue de données ont ceci de particulier qu'elles sont classées dans la catégorie des vues de définisseur, où les autorisations d'accès sont définies par l'utilisateur qui a créé la vue et non par l'utilisateur qui interroge la vue. Voici quelques cas d'utilisation et avantages de la création de vues dans le catalogue de données :

- Créer une vue qui restreint l'accès aux données en fonction des autorisations dont l'utilisateur a besoin. Par exemple, vous pouvez utiliser des vues du catalogue de données pour empêcher les employés qui ne travaillent pas dans le service des ressources humaines (RH) de voir des données d'identification personnelle (PII).
- S'assurer que les utilisateurs ne peuvent pas accéder aux enregistrements incomplets. En appliquant certains filtres aux vues du catalogue de données, vous vous assurez que les enregistrements de données d'une vue du catalogue de données sont toujours complets.
- Les vues du catalogue de données présentent un avantage en matière de sécurité, car la définition de requête utilisée pour créer la vue doit être complète pour que la vue soit créée. Cet avantage en matière de sécurité signifie que les vues du catalogue de données ne sont pas sensibles aux commandes SQL provenant d'acteurs malveillants.
- Les vues du catalogue de données présentent les mêmes avantages que les vues normales, notamment en permettant aux utilisateurs d'accéder à une vue sans mettre la table sous-jacente à leur disposition.

Pour créer une vue dans le catalogue de données, vous devez disposer d'une table externe Spectrum, d'un objet contenu dans une unité de partage des données gérée par Lake Formation ou d'une table Apache Iceberg.

Les définitions des vues du catalogue de données sont stockées dans le AWS Glue Data Catalog. AWS Lake Formation À utiliser pour accorder l'accès via des autorisations de ressources, des

autorisations de colonnes ou des contrôles d'accès basés sur des balises. Pour plus d'informations sur l'octroi et la révocation de l'accès dans Lake Formation, consultez Octroi et révocation d'autorisations liées aux ressources du catalogue de données.

Prérequis

Avant de pouvoir créer une vue dans le catalogue de données, assurez-vous que les conditions préalables suivantes sont remplies :

- Assurez-vous que votre rôle IAM possède la stratégie d'approbation suivante.

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

- Vous avez aussi besoin de la stratégie de transmission de rôle suivante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1",
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
```

```
                        "glue.amazonaws.com",
                        "lakeformation.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

- Enfin, vous aurez également besoin des autorisations suivantes.
  - `Glue:GetDatabase`
  - `Glue:GetDatabases`
  - `Glue:CreateTable`
  - `Glue:GetTable`
  - `Glue:UpdateTable`
  - `Glue:DeleteTable`
  - `Glue:GetTables`
  - `Glue:SearchTables`
  - `Glue:BatchGetPartition`
  - `Glue:GetPartitions`
  - `Glue:GetPartition`
  - `Glue:GetTableVersion`
  - `Glue:GetTableVersions`

End-to-end exemple

Commencez par créer un schéma externe basé sur la base de données de votre catalogue de données.

```
CREATE EXTERNAL SCHEMA IF NOT EXISTS external_schema FROM DATA CATALOG DATABASE
 'external_data_catalog_db'
IAM_ROLE 'arn:aws:iam::123456789012:role/sample-role';
```

Vous pouvez désormais créer une vue du catalogue de données.

```
CREATE EXTERNAL PROTECTED VIEW external_schema.remote_view
```

```
AS SELECT * FROM external_schema.remote_table;
```

Vous pouvez ensuite commencer à interroger cette vue.

```
SELECT * FROM external_schema.remote_view;
```

Pour plus d'informations sur les commandes SQL associées aux vues du catalogue de données, consultez CREATE EXTERNAL VIEW, ALTER EXTERNAL VIEW et DROP EXTERNAL VIEW.

Journalisation sécurisée

Redshift masque les métadonnées enregistrées dans les journaux du système Redshift lorsqu'une requête fait référence à une vue collée multidialectale. Le multidialecte signifie que la vue prend en charge les dialectes SQL de différents moteurs de requête, tels que Redshift et Amazon EMR. Les données des tables suivantes sont masquées pour toutes les requêtes ayant le même identifiant de requête. Le tableau suivant répertorie les vues et les colonnes du système auxquelles la journalisation sécurisée est appliquée.

Tableau du système	Colonnes sensibles
SYS_EXTERNAL_QUERY_DETAIL	Colonnes : source_type, total_partitions, qualified_partitions, scanned_files, returned_rows, returned_bytes, file_format, file_location, external_query_text, warning_message. Pour de plus amples informations, veuillez consulter SYS_EXTERNAL_QUERY_DETAIL.
SYS_EXTERNAL_QUERY_ERROR	Colonnes : file_location, rowid, column_name, original_value, modified_value, trigger, action, action_value, error_code. Pour de plus amples informations, veuillez consulter SYS_EXTERNAL_QUERY_ERROR.
SYS_QUERY_DETAIL	Colonnes : step_name, table_id, table_name, input_bytes, input_rows, output_bytes, output_rows, blocks_read, blocks_write, local_read_io, remote_read_io, spilled_block_local_disk, spilled_block_remote_disk. Pour de plus amples informations, veuillez consulter SYS_QUERY_DETAIL.
SYS_QUERY_HISTORY	Colonnes : returned_rows, returned_bytes. Pour de plus amples informations, veuillez consulter SYS_QUERY_HISTORY.

Tableau du système	Colonnes sensibles
STL_AGGR	Colonnes : lignes, octets, tbl, type. Pour de plus amples informations, veuillez consulter STL_AGGR.
STL_BCAST	Colonnes : lignes, octets, paquets. Pour de plus amples informations, veuillez consulter STL_BCAST.
STL_DDLTEXT	Colonnes : texte Pour de plus amples informations, veuillez consulter STL_DDLTEXT.
STL_DELETE	Colonnes : lignes, tabl. Pour de plus amples informations, veuillez consulter STL_DELETE.
STL_DIST	Colonnes : lignes, octets, paquets. Pour de plus amples informations, veuillez consulter STL_DIST.
STL_EXPLAIN	Colonnes : plannode, info. Pour de plus amples informations, veuillez consulter STL_EXPLAIN.
STL_HASH	Colonnes : lignes, octets, tbl, est_rows. Pour de plus amples informati ons, veuillez consulter STL_HASH.
STL_HASHJOIN	Colonnes : rows, tbl, num_parts, join_type. Pour de plus amples informations, veuillez consulter STL_HASHJOIN.
STL_INSERT	Colonnes : lignes, tabl. Pour de plus amples informations, veuillez consulter STL_INSERT.
STL_LIMIT	Colonnes : lignes Pour de plus amples informations, veuillez consulter STL_LIMIT.
STL_MERGE	Colonnes : lignes Pour de plus amples informations, veuillez consulter STL_MERGE.
STL_MERGEJOIN	Colonnes : lignes, tabl. Pour de plus amples informations, veuillez consulter STL_MERGEJOIN.
STL_NESTLOOP	Colonnes : lignes, tabl. Pour de plus amples informations, veuillez consulter STL_NESTLOOP.

Tableau du système	Colonnes sensibles
STL_PARSE	Colonnes : lignes Pour de plus amples informations, veuillez consulter STL_PARSE.
STL_PLAN_INFO	Colonnes : lignes, octets. Pour de plus amples informations, veuillez consulter STL_PLAN_INFO.
STL_PROJECT	Colonnes : lignes, tabl. Pour de plus amples informations, veuillez consulter STL_PROJECT.
STL_QUERY	Colonnes : querytxt. Pour de plus amples informations, veuillez consulter STL_QUERY.
STL_QUERY_METRICS	Colonnes : max_rows, rows, max_blocks_read, blocks_read, max_blocks_to_disk, blocks_to_disk, max_query_scan_size, query_scan_size. Pour de plus amples informations, veuillez consulter STL_QUERY_METRICS.
STL_QUERYTEXT	Colonnes : texte Pour de plus amples informations, veuillez consulter STL_QUERYTEXT.
STL_RETURN	Colonnes : lignes, octets. Pour de plus amples informations, veuillez consulter STL_RETURN.
STL_SAVE	Colonnes : lignes, octets, tbl. Pour de plus amples informations, veuillez consulter STL_SAVE.
STL_SCAN	Colonnes : lignes, octets, extractions, type, tbl, rows_pre_filter, perm_table_name, scanned_mega_value. Pour de plus amples informations, veuillez consulter STL_SCAN.
STL_SORT	Colonnes : lignes, octets, tbl. Pour de plus amples informations, veuillez consulter STL_SORT.
STL_TR_CONFLICT	Colonnes : table_id. Pour de plus amples informations, veuillez consulter STL_TR_CONFLICT.

Tableau du système	Colonnes sensibles
STL_UNDONE	Colonnes : table_id. Pour de plus amples informations, veuillez consulter STL_UNDONE.
STL_UNIQUE	Colonnes : lignes, type, octets. Pour de plus amples informations, veuillez consulter STL_UNIQUE.
STL_UTILITYTEXT	Colonnes : texte Pour de plus amples informations, veuillez consulter STL_UTILITYTEXT.
STL_WINDOW	Colonnes : lignes Pour de plus amples informations, veuillez consulter STL_WINDOW.
STV_BLOCKLIST	Colonnes : col, tbl, num_values, minvalue, maxvalue. Pour de plus amples informations, veuillez consulter STV_BLOCKLIST.
STV_EXEC_STATE	Colonnes : lignes, octets, étiquette. Pour de plus amples informations, veuillez consulter STV_EXEC_STATE.
STV_LOCKS	Colonnes : table_id. Pour de plus amples informations, veuillez consulter STV_LOCKS.
STV_QUERY_METRICS	Colonnes : lignes, max_rows, blocks_read, max_blocks_read, max_blocks_to_disk, blocks_to_disk, max_query_scan_size, query_scan_size. Pour de plus amples informations, veuillez consulter STV_QUERY_METRICS.
STV_STARTUP_RECOVERY_STATE	Colonnes : table_id, table_name. Pour de plus amples informations, veuillez consulter STV_STARTUP_RECOVERY_STATE.
STV_TBL_PERM	Colonnes : id, rows, sorted_rows, temp, block_count, query_scan_size. Pour de plus amples informations, veuillez consulter STV_TBL_PERM.
STV_TBL_TRANS	Colonnes : identifiant, lignes, taille. Pour de plus amples informations, veuillez consulter STV_TBL_TRANS.

Tableau du système	Colonnes sensibles
SVCS_EXPLAIN	Colonnes : plannode, info. Pour de plus amples informations, veuillez consulter SVCS_EXPLAIN.
SVCS_PLAN_INFO	Colonnes : lignes, octets. Pour de plus amples informations, veuillez consulter SVCS_PLAN_INFO.
SVCS_QUERY_SUMMARY	Colonnes : step, rows, bytes, rate_row, rate_byte, label, rows_pre_filter. Pour de plus amples informations, veuillez consulter SVCS_QUERY_SUMMARY.
SVCS_S3LIST	Colonnes : bucket, prefix, max_file_size, avg_file_size. Pour de plus amples informations, veuillez consulter SVCS_QUERY_SUMMARY.
SVCS_S3LOG	Colonnes : message Pour de plus amples informations, veuillez consulter SVCS_QUERY_SUMMARY.
SVCS_S3PARTITION_SUMMARY	Colonnes : total_partitions, qualified_partitions, min_assigned_partitions, max_assigned_partitions, avg_assigned_partitions. Pour de plus amples informations, veuillez consulter SVCS_S3PARTITION_SUMMARY.
SVCS_S3QUERY_SUMMARY	Colonnes : external_table_name, file_format, s3_scanned_rows, s3_scanned_bytes, s3query_returned_rows, s3query_returned_bytes. Pour de plus amples informations, veuillez consulter SVCS_S3QUERY_SUMMARY.
SVL_QUERY_METRICS	Colonnes : step_label, scan_row_count, join_row_count, nested_loop_join_row_count, return_row_count, spectrum_scan_row_count, spectrum_scan_size_mb. Pour de plus amples informations, veuillez consulter SVL_QUERY_METRICS.
SVL_QUERY_METRICS_SUMMARY	Colonnes : step_label, scan_row_count, join_row_count, nested_loop_join_row_count, return_row_count, spectrum_scan_row_count, spectrum_scan_size_mb. Pour de plus amples informations, veuillez consulter SVL_QUERY_METRICS_SUMMARY.

Tableau du système	Colonnes sensibles
SVL_QUERY_REPORT	Colonnes : lignes, octets, label, rows_pre_filter. Pour de plus amples informations, veuillez consulter SVL_QUERY_REPORT.
SVL_QUERY_SUMMARY	Colonnes : lignes, octets, rows_pre_filter. Pour de plus amples informations, veuillez consulter SVL_QUERY_SUMMARY.
SVL_S3LIST	Colonnes : bucket, prefix, max_file_size, avg_file_size. Pour de plus amples informations, veuillez consulter SVL_S3LIST.
SVL_S3LOG	Colonnes : message Pour de plus amples informations, veuillez consulter SVL_S3LOG.
SVL_S3PARTITION	Colonnes : lignes, octets, label, rows_pre_filter. Pour de plus amples informations, veuillez consulter SVL_S3PARTITION.
SVL_S3PARTITION_SUMMARY	Colonnes : total_partitions, qualified_partitions, min_assigned_partitions, max_assigned_partitions, avg_assigned_partitions. Pour de plus amples informations, veuillez consulter SVL_S3PARTITION_SUMMARY.
SVL_S3QUERY	Colonnes : external_table_name, file_format, s3_scanned_rows, s3_scanned_bytes, s3query_returned_rows, s3query_returned_bytes. Pour de plus amples informations, veuillez consulter SVL_S3QUERY.
SVL_S3QUERY_SUMMARY	Colonnes : external_table_name, file_format, s3_scanned_rows, s3_scanned_bytes, s3query_returned_rows, s3query_returned_bytes. Pour de plus amples informations, veuillez consulter SVL_S3QUERY_SUMMARY.
SVL_S3RETRIES	Colonnes : file_size, location, message. Pour de plus amples informations, veuillez consulter SVL_S3RETRIES.
SVL_SPECTRUM_SCAN_ERROR	Colonnes : location, rowid, colname, original_value, modified_value. Pour de plus amples informations, veuillez consulter SVL_SPECTRUM_SCAN_ERROR.

Tableau du système	Colonnes sensibles
SVL_STATEMENTTEXT	Colonnes : type, texte Pour de plus amples informations, veuillez consulter SVL_STATEMENTTEXT.
SVL_STORED_PROC_CALL	Colonnes : querytxt. Pour de plus amples informations, veuillez consulter SVL_STORED_PROC_CALL.
SVL_STORED_PROC_MESSAGES	Colonnes : texte de requête. Pour de plus amples informations, veuillez consulter SVL_STORED_PROC_MESSAGES.
SVL_UDF_LOG	Colonnes : funcname. Pour de plus amples informations, veuillez consulter SVL_UDF_LOG.
SVV_DISKUSAGE	Colonnes : name, col, tbl, blocknum, num_values, minvalue, maxvalue. Pour de plus amples informations, veuillez consulter SVV_DISKUSAGE.
SVV_QUERY_STATE	Colonnes : lignes, octets, étiquette. Pour de plus amples informations, veuillez consulter SVV_QUERY_STATE.
SVV_TABLE_INFO	Colonnes : table_id, table. Pour de plus amples informations, veuillez consulter SVV_TABLE_INFO.
SVV_TRANSACTIONS	Colonnes : relation. Pour de plus amples informations, veuillez consulter SVV_TRANSACTIONS.

Considérations et restrictions

Les considérations et restrictions suivantes s'appliquent aux vues créées dans le catalogue de données.

- AWS Glue Data Catalog les vues ne sont prises en charge que sur les clusters RA3 provisionnés ou les groupes de travail Redshift Serverless.
- Vous ne pouvez pas créer une vue du catalogue de données basée sur une autre vue.
- Une vue du catalogue de données ne peut comporter que 10 tables de base.
- Le définisseur de la vue doit disposer d'autorisations `SELECT GRANTABLE` complètes sur les tables de base.

- Les vues ne peuvent contenir que des éléments intégrés et des objets Lake Formation. Les objets suivants ne sont pas autorisés à l’intérieur d’une vue.
  - Tables système
  - Fonctions définies par l'utilisateur () UDFs
  - Tables, vues, vues matérialisées et vues à liaison tardive Redshift qui ne se trouvent pas dans une unité de partage des données gérée par Lake Formation.
- Les vues ne peuvent pas contenir de tables Redshift Spectrum imbriquées.
- AWS Glue les représentations des objets de base d'une vue doivent être dans la même Compte AWS région que la vue.

Interrogation des données spatiales dans Amazon Redshift

Les données spatiales décrivent la position et la forme d'une géométrie dans un espace défini (un système de référence spatial). Amazon Redshift prend en charge les données spatiales avec les types de données GEOMETRY et GEOGRAPHY, qui contiennent des données spatiales et éventuellement l'identifiant de système de référence spatiale (SRID) des données.

Les données spatiales contiennent des données géométriques qui peuvent être utilisées pour représenter des fonctions géographiques. Les rapports météo, les indications de direction, les tweets intégrant des lieux géographiques, les emplacements de magasins et les lignes aériennes sont des exemples de ce type de données. Les données spatiales jouent un rôle important dans les analyses métier, la création de rapports et les prévisions.

Vous pouvez interroger des données spatiales avec les fonctions SQL Amazon Redshift. Les données spatiales contiennent des valeurs géométriques pour un objet.

Les opérations de type de données GEOMETRY fonctionnent sur le plan cartésien. Bien que l'identifiant du système de référence spatiale (SRID) soit stocké à l'intérieur de l'objet, ce SRID n'est qu'un identifiant du système de coordonnées et ne joue aucun rôle dans les algorithmes utilisés pour traiter les objets GEOMETRY. À l'inverse, les opérations sur le type de données GEOGRAPHY traitent les coordonnées à l'intérieur des objets comme des coordonnées sphériques sur un sphéroïde. Ce sphéroïde est défini par le SRID, qui fait référence à un système de référence spatiale géographique. Par défaut, les types de données GEOGRAPHY sont créés avec la référence spatiale (SRID) 4326, faisant référence au système géodésique mondial (WGS) 84. Pour plus d'informations SRIDs, voir Système de référence spatiale sur Wikipedia.

Vous pouvez utiliser la fonction ST_Transform pour transformer les coordonnées de différents systèmes de référence spatiale. Une fois la transformation des coordonnées terminée, vous pouvez également utiliser une simple distribution entre les deux, à condition que l'entrée GEOMETRY soit codée avec le SRID géographique. Cette conversion copie simplement les coordonnées sans aucune autre transformation. Par exemple :

```
SELECT ST_AsEWKT(ST_GeomFromEWKT('SRID=4326;POINT(10 20)')::geography);
```

```
st_asewkt
------------------------
 SRID=4326;POINT(10 20)
```

Pour mieux comprendre la différence entre les types de données GEOMETRY et GEOGRAPHY, envisagez de calculer la distance entre l'aéroport de Berlin (BER) et l'aéroport de San Francisco (SFO) à l'aide du système géodésique mondial (WGS) 84. En utilisant le type de données GEOGRAPHY, le résultat est exprimé en mètres. Lorsque vous utilisez le type de données GEOMETRY avec SRID 4326, le résultat est exprimé en degrés, qui ne peuvent pas être convertis en mètres, car la distance d'un degré dépend de l'emplacement des géométries du globe.

Les calculs sur les types de données GEOGRAPHY sont principalement utilisés pour des calculs réalistes de terre ronde, tels que la zone précise d'un pays sans distorsion. Cependant, leur calcul coûte beaucoup plus cher. Par conséquent, ST_Transform peut transformer vos coordonnées en un système de coordonnées projetées local approprié et effectuer le calcul sur le type de données GEOMETRY plus rapidement.

À l'aide des données spatiales, vous pouvez exécuter des requêtes afin d'effectuer les opérations suivantes :

- Trouver la distance entre deux points.
- Vérifier si une zone (polygone) en contient une autre.
- Vérifier si une linestring coupe une autre linestring ou un polygone.

Vous pouvez utiliser le type de données GEOMETRY pour contenir les valeurs des données spatiales. Une valeur GEOMETRY dans Amazon Redshift peut définir des types de données primitifs de géométrie bidimensionnelle (2D), tridimensionnelle (3DZ), bidimensionnelle avec une mesure (3DM) et quadridimensionnelle (4D) :

- Une géométrie bidimensionnelle (2D) est spécifiée par deux coordonnées cartésiennes (x, y) dans un plan.
- Une géométrie tridimensionnelle (3DZ) est spécifiée par trois coordonnées cartésiennes (x, y, z) dans l'espace.
- Une géométrie bidimensionnelle avec mesure (3DM) est spécifiée par trois coordonnées (x, y, m), où les deux premières sont des coordonnées cartésiennes dans un plan et la troisième est une mesure.
- Une géométrie quadridimensionnelle (4D) est spécifiée par quatre coordonnées (x, y, z, m), où les trois premières sont des coordonnées cartésiennes dans un espace et la quatrième est une mesure.

Pour plus d'informations sur les types de données primitives géométriques, consultez Well-known text dans Wikipedia.

Vous pouvez utiliser le type de données GEOGRAPHY pour contenir les valeurs des données spatiales. Une valeur GEOGRAPHY dans Amazon Redshift peut définir des types de données primitifs de géométrie bidimensionnelle (2D), tridimensionnelle (3DZ), bidimensionnelle avec une mesure (3DM) et quadridimensionnelle (4D) :

- Une géométrie bidimensionnelle (2D) est spécifiée par des coordonnées de longitude et de latitude sur un sphéroïde.
- Une géométrie tridimensionnelle (3DZ) est spécifiée par des coordonnées de longitude, de latitude et d'altitude sur un sphéroïde.
- Une géométrie bidimensionnelle avec mesure (3DM) est spécifiée par trois coordonnées (longitude, latitude et mesure), où les deux premières sont des coordonnées angulaires sur une sphère et la troisième est une mesure.
- Une géométrie quadridimensionnelle (4D) est spécifiée par quatre coordonnées (longitude, latitude, altitude, mesure), où les trois premières sont la longitude, la latitude et l'altitude, et la quatrième est une mesure.

Pour plus d'informations sur les systèmes de coordonnées géographiques, consultez Coordonnées géographiques et Coordonnées sphériques sur Wikipédia.

Les types de données GEOMETRY et GEOGRAPHY ont les sous-types suivants :

- POINT
- LINESTRING
- POLYGON
- MULTIPOINT
- MULTILINESTRING
- MULTIPOLYGON
- GEOMETRYCOLLECTION

Des fonctions SQL Amazon Redshift prennent en charge les représentations suivantes des données géométriques :

- GeoJSON

- Well-known text (WKT)
- Extended well-known text (EWKT)
- Représentation WKB (Well-known binary)
- Extended well-known binary (EWKB)

Vous pouvez effectuer une conversion entre les types de données GEOMETRY et GEOGRAPHY.

Le code SQL suivant convertit une linestring à partir d'une GEOMETRY en une GEOGRAPHY.

```
SELECT ST_AsEWKT(ST_GeomFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)')::geography);
```

```
 st_asewkt
---------------------------------------------
 SRID=4326;LINESTRING(110 40,2 3,-10 80,-7 9)
```

Le code SQL suivant convertit une linestring à partir d'une GEOGRAPHY en une GEOMETRY.

```
SELECT ST_AsEWKT(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)')::geometry);
```

```
 st_asewkt
---------------------------------------------
 SRID=4326;LINESTRING(110 40,2 3,-10 80,-7 9)
```

Amazon Redshift fournit de nombreuses fonctions SQL pour interroger les données spatiales. À l'exception de la fonction ST_IsValid, les fonctions spatiales qui acceptent un objet GEOMETRY comme argument s'attendent à ce que cet objet GEOMETRY soit une géométrie valide. Si l'objet GEOMETRY ou GEOGRAPHY n'est pas valide, le comportement de la fonction spatiale n'est pas défini. Pour plus d'informations sur la validité, consultez Validité géométrique.

Pour plus d'informations sur les fonctions SQL permettant d'interroger les données spatiales, consultez Fonctions spatiales.

Pour plus d'informations sur le chargement des données spatiales, consultez Chargement d'une colonne avec le type de données GEOMETRY ou GEOGRAPHY.

Rubriques

- Didacticiel : Utiliser les fonctions SQL spatiales avec Amazon Redshift
- Chargement d'un shapefile dans Amazon Redshift
- Terminologie des données spatiales Amazon Redshift
- Éléments à prendre en compte lors de l'utilisation de données spatiales dans Amazon Redshift

Didacticiel : Utiliser les fonctions SQL spatiales avec Amazon Redshift

Ce didacticiel explique comment utiliser certaines fonctions SQL spatiales avec Amazon Redshift

Pour ce faire, vous interrogez deux tables à l'aide de fonctions SQL spatiales. Le didacticiel utilise des données provenant de jeux de données publics qui mettent en corrélation les données de localisation des logements locatifs avec des codes postaux à Berlin, en Allemagne.

Rubriques

- Prérequis
- Étape 1 : Créer des tables et charger des données de test
- Étape 2 : Interroger les données spatiales
- Étape 3 : Nettoyer vos ressources

Prérequis

Pour ce didacticiel, vous avez besoin des ressources suivantes :

- Un cluster et une base de données Amazon Redshift existants auxquels vous pouvez accéder et que vous pouvez mettre à jour. Dans le cluster existant, vous créez des tables, chargez des exemples de données et exécutez des requêtes SQL pour démontrer les fonctions spatiales. Votre cluster doit disposer d'au moins deux nœuds. Pour savoir comment créer un cluster, suivez les étapes du Guide de démarrage d'Amazon Redshift.
- Pour utiliser l'éditeur de requêtes Amazon Redshift, assurez-vous que votre cluster se trouve dans une région AWS qui prend en charge l'éditeur de requêtes. Pour plus d'informations, consultez Interrogation d'une base de données à l'aide de l'éditeur de requêtes dans le Guide de gestion Amazon Redshift.
- AWS informations d'identification pour votre cluster Amazon Redshift qui lui permettent de charger des données de test depuis Amazon S3. Pour plus d'informations sur la manière d'accéder à

d'autres AWS services tels qu'Amazon S3, consultez Autoriser Amazon Redshift à accéder aux AWS services.

- Rôle AWS Identity and Access Management (IAM) nommémySpatialDemoRole, auquel est AmazonS3ReadOnlyAccess attachée la politique gérée pour lire les données Amazon S3. Pour créer un rôle avec l'autorisation de charger les données à partir d'un compartiment Amazon S3, consultez Autorisation d'opérations COPY, UNLOAD, CREATE EXTERNAL FUNCTION et CREATE EXTERNAL SCHEMA à l'aide de rôles IAM dans le Guide de gestion Amazon Redshift.
- Une fois que vous avez créé le rôle IAM mySpatialDemoRole, celui-ci nécessite une association avec votre cluster Amazon Redshift. Pour plus d'informations sur la création d'une association, consultez Autorisation d'opérations COPY, UNLOAD, CREATE EXTERNAL FUNCTION et CREATE EXTERNAL SCHEMA à l'aide de rôles IAM dans le Guide de gestion Amazon Redshift.

Étape 1 : Créer des tables et charger des données de test

Les données source utilisées par ce didacticiel se trouvent dans des fichiers nommés accommodations.csv et zipcodes.csv.

Le fichier accommodations.csv comprend des données open source provenant de insideairbnb.com. Le fichier zipcodes.csv fournit des codes postaux qui sont des données open source de l'institut national de statistique de Berlin-Brandebourg en Allemagne (Amt für Statistik Berlin-Brandebourg). Les deux sources de données sont fournies sous licence Creative Commons. Les données sont limitées à la région de Berlin, en Allemagne. Ces fichiers se trouvent dans un compartiment public Amazon S3 à utiliser avec ce didacticiel.

Vous pouvez éventuellement télécharger les données source à partir des liens Amazon S3 suivants :

- Données source pour la table accommodations.
- Données source pour la table zipcode.

Utilisez la procédure suivante pour créer des tables et charger les données de test.

Pour créer des tables et charger les données de test

1. Ouvrez l'éditeur de requêtes Amazon Redshift. Pour plus d'informations sur l'utilisation de l'éditeur de requêtes, consultez Interrogation d'une base de données à l'aide de l'éditeur de requêtes dans le Guide de la gestion du cluster Amazon Redshift.

2. Supprimez toutes les tables utilisées par ce didacticiel si elles existent déjà dans votre base de données. Pour plus d'informations, consultez Étape 3 : Nettoyer vos ressources.

3. Créez la table `accommodations` pour stocker l'emplacement géographique de chaque hébergement (longitude et latitude), le nom de la liste et d'autres données métier.

   Ce didacticiel parcourt les locations de chambres à Berlin, en Allemagne. La colonne `shape` stocke les points géographiques de l'emplacement des logements. Les autres colonnes contiennent des informations sur la location.

   Pour créer la table `accommodations`, exécutez l'instruction SQL suivante dans l'éditeur de requêtes Amazon Redshift.

   ```
   CREATE TABLE public.accommodations (
     id INTEGER PRIMARY KEY,
     shape GEOMETRY,
     name VARCHAR(100),
     host_name VARCHAR(100),
     neighbourhood_group VARCHAR(100),
     neighbourhood VARCHAR(100),
     room_type VARCHAR(100),
     price SMALLINT,
     minimum_nights SMALLINT,
     number_of_reviews SMALLINT,
     last_review DATE,
     reviews_per_month NUMERIC(8,2),
     calculated_host_listings_count SMALLINT,
     availability_365 SMALLINT
   );
   ```

4. Créez la table `zipcode` dans l'éditeur de requêtes pour stocker les codes postaux de Berlin.

   Un code postal est défini en tant que polygone dans la colonne `wkb_geometry`. Les autres colonnes décrivent les métadonnées spatiales supplémentaires sur le code postal.

   Pour créer la table `zipcode`, exécutez l'instruction SQL suivante dans l'éditeur de requêtes Amazon Redshift.

   ```
   CREATE TABLE public.zipcode (
     ogc_field INTEGER PRIMARY KEY NOT NULL,
     wkb_geometry GEOMETRY,
     gml_id VARCHAR(256),
   ```

```
  spatial_name VARCHAR(256),
  spatial_alias VARCHAR(256),
  spatial_type VARCHAR(256)
 );
```

5. Chargez les tables à l'aide des exemples de données.

   Les exemples de données de ce didacticiel sont fournis dans un compartiment Amazon S3 qui permet un accès en lecture à tous les AWS utilisateurs authentifiés. Assurez-vous que vous fournissez les informations d'identification AWS valides qui permettent d'accéder à Amazon S3.

   Pour charger les données de test dans vos tables, exécutez les commandes COPY suivantes. Remplacez *account-number* par votre propre numéro de compte AWS . Le segment de la chaîne d'informations d'identification entre guillemets simples ne peut pas comporter d'espaces ou de sauts de ligne.

```
COPY public.accommodations
FROM 's3://redshift-downloads/spatial-data/accommodations.csv'
DELIMITER ';'
IGNOREHEADER 1 REGION 'us-east-1'
CREDENTIALS 'aws_iam_role=arn:aws:iam::account-number:role/mySpatialDemoRole';
```

```
COPY public.zipcode
FROM 's3://redshift-downloads/spatial-data/zipcode.csv'
DELIMITER ';'
IGNOREHEADER 1 REGION 'us-east-1'
CREDENTIALS 'aws_iam_role=arn:aws:iam::account-number:role/mySpatialDemoRole';
```

6. Vérifiez que chaque table s'est chargée correctement en exécutant les commandes suivantes.

```
select count(*) from accommodations;
```

```
select count(*) from zipcode;
```

   Les résultats suivants indiquent le nombre de lignes dans chaque tableau de données de test.

Nom de la table	Lignes
Hébergements	22 248

Nom de la table	Lignes
zipcode	190

Étape 2 : Interroger les données spatiales

Une fois vos tables créées et chargées, vous pouvez les interroger à l'aide des instructions SQL SELECT. Les requêtes suivantes illustrent certaines des informations que vous pouvez récupérer. Vous pouvez écrire de nombreuses autres requêtes qui utilisent des fonctions spatiales pour répondre à vos besoins.

Pour interroger les données spatiales

1. Interrogez les données pour obtenir le nombre total d'éléments stockés dans la table `accommodations`, comme illustré ci-dessous. Le système de référence spatiale est le système géodésique mondial (WGS) 84, qui possède l'identifiant de référence spatiale unique 4326.

```
SELECT count(*) FROM public.accommodations WHERE ST_SRID(shape) = 4326;
```

```
 count
-------
 22248
```

2. Extrayez les objets géométriques au format WKT (Well-known text) avec quelques attributs en plus. En outre, vous pouvez vérifier si ces données de code postal sont également stockées dans le système géodésique mondial (WGS) 84, qui utilise l'ID de référence spatiale (SRID) 4326. Les données spatiales doivent être stockées dans le même système de référence spatiale pour être interopérables.

```
SELECT ogc_field, spatial_name, spatial_type, ST_SRID(wkb_geometry),
 ST_AsText(wkb_geometry)
FROM public.zipcode
ORDER BY spatial_name;
```

```
ogc_field  spatial_name  spatial_type  st_srid  st_astext
-----------------------------------------------------------------
```

```
0           10115        Polygon        4326     POLYGON((...))
4           10117        Polygon        4326     POLYGON((...))
8           10119        Polygon        4326     POLYGON((...))
...
(190 rows returned)
```

3. Sélectionnez le polygone de Berlin Mitte (10117), un arrondissement de Berlin, au format GeoJSON, sa dimension et le nombre de points dans ce polygone.

```
SELECT ogc_field, spatial_name, ST_AsGeoJSON(wkb_geometry),
 ST_Dimension(wkb_geometry), ST_NPoints(wkb_geometry)
FROM public.zipcode
WHERE spatial_name='10117';
```

```
ogc_field  spatial_name  spatial_type
 st_dimension  st_npoint
--------------------------------------------------------------------------------------------
4           10117         {"type":"Polygon", "coordinates":[[[...]]]}    2
    331
```

4. Exécutez la commande SQL suivante pour voir combien d'hébergements se trouvent à moins de 500 mètres de la porte de Brandebourg.

```
SELECT count(*)
FROM public.accommodations
WHERE ST_DistanceSphere(shape, ST_GeomFromText('POINT(13.377704 52.516431)', 4326))
 < 500;
```

```
count
------
  29
```

5. Obtenez l'emplacement approximatif de la porte de Brandebourg à partir des données stockées dans les hébergements répertoriés comme étant à proximité en exécutant la requête suivante.

   Cette requête nécessite une sous-sélection. Le nombre obtenu n'est pas le même parce que l'emplacement interrogé diffère de la requête précédente, car il est plus proche des hébergements.

```
WITH poi(loc) as (
  SELECT st_astext(shape) FROM accommodations WHERE name LIKE '%brandenburg gate%'
)
SELECT count(*)
FROM accommodations a, poi p
WHERE ST_DistanceSphere(a.shape, ST_GeomFromText(p.loc, 4326)) < 500;
```

```
count
------
  60
```

6. Exécutez la requête suivante pour afficher les détails de tous les hébergements autour de la porte de Brandebourg, classés par prix dans l'ordre décroissant.

```
SELECT name, price, ST_AsText(shape)
FROM public.accommodations
WHERE ST_DistanceSphere(shape, ST_GeomFromText('POINT(13.377704 52.516431)', 4326))
 < 500
ORDER BY price DESC;
```

```
name                                                    price  st_astext
-------------------------------------------------------------------------------------------
DUPLEX APARTMENT/PENTHOUSE in 5* LOCATION! 7583         300
 POINT(13.3826510209548 52.5159819722552)
DUPLEX-PENTHOUSE IN FIRST LOCATION! 7582                300
 POINT(13.3799997083855 52.5135918444834)
...
(29 rows returned)
```

7. Exécutez la requête suivante pour récupérer l'hébergement le plus cher avec son code postal.

```
SELECT
  a.price, a.name, ST_AsText(a.shape),
  z.spatial_name, ST_AsText(z.wkb_geometry)
FROM accommodations a, zipcode z
WHERE price = 9000 AND ST_Within(a.shape, z.wkb_geometry);
```

```
price   name                                st_astext
     spatial_name      st_astext
-------------------------------------------------------------------------------------------
9000    Ueber den Dächern Berlins Zentrum    POINT(13.334436985013
 52.4979779501538)    10777             POLYGON((13.3318284987227
 52.4956021172799,...
```

8. Calculez le prix maximum, minimum ou médian des hébergements à l'aide d'une sous-requête.

   La requête suivante répertorie le prix médian des logements par code postal.

```
SELECT
  a.price, a.name, ST_AsText(a.shape),
  z.spatial_name, ST_AsText(z.wkb_geometry)
FROM accommodations a, zipcode z
WHERE
  ST_Within(a.shape, z.wkb_geometry) AND
  price = (SELECT median(price) FROM accommodations)
ORDER BY a.price;
```

```
price name                            st_astext
 spatial_name   st_astext
-------------------------------------------------------------------------------------------

45    "Cozy room Berlin-Mitte"        POINT(13.3864349535358 52.5292016386514)
 10115          POLYGON((13.3658598465795 52.535659581048,...
...
(723 rows returned)
```

9. Exécutez la requête suivante pour récupérer le nombre d'hébergements répertoriés à Berlin. Les points chauds sont regroupés par code postal et triés par quantité d'informations fournies.

```
SELECT z.spatial_name as zip, count(*) as numAccommodations
FROM public.accommodations a, public.zipcode z
WHERE ST_Within(a.shape, z.wkb_geometry)
GROUP BY zip
ORDER BY numAccommodations DESC;
```

```
zip   numaccommodations
---------------------------
```

```
10245 872
10247 832
10437 733
10115 664
...
(187 rows returned)
```

Étape 3 : Nettoyer vos ressources

Votre cluster continue d'accumuler les frais aussi longtemps qu'il est en cours d'exécution. Lorsque vous aurez terminé ce didacticiel, vous pourrez supprimer votre exemple de cluster.

Si vous souhaitez conserver le cluster, mais récupérer le stockage utilisé par les tables de données de test, exécutez les commandes suivantes pour supprimer les tables.

```
drop table public.accommodations cascade;
```

```
drop table public.zipcode cascade;
```

Chargement d'un shapefile dans Amazon Redshift

Vous pouvez utiliser la commande COPY pour ingérer des shapefiles Esri stockés dans Amazon S3 dans des tables Amazon Redshift. Un shapefile stocke l'emplacement géométrique et les informations d'attribut des fonctions géographiques dans un format vectoriel. Le format du shapefile peut décrire les aspects spatiaux des objets spatiaux tels que les points, les lignes et les polygones. Pour plus d'informations sur les shapefiles, consultez Shapefile dans Wikipédia.

La commande COPY prend en charge le paramètre de format de données `SHAPEFILE`. Par défaut, la première colonne du shapefile est une colonne `GEOMETRY` ou `IDENTITY`. Toutes les colonnes suivantes suivent l'ordre spécifié dans le shapefile. Cependant, la table cible n'a pas besoin d'être dans cette disposition exacte, car vous pouvez utiliser le mappage de colonne COPY pour définir l'ordre. Pour plus d'informations sur la prise en charge du shapefile de commande COPY, consultez SHAPEFILE.

Dans certains cas, la taille de géométrie résultante peut être supérieure au maximum pour stocker une géométrie dans Amazon Redshift. Si c'est le cas, vous pouvez utiliser l'option COPY `SIMPLIFY` ou `SIMPLIFY AUTO` pour simplifier les géométries pendant l'ingestion comme suit :

- Spécifiez `SIMPLIFY` *`tolerance`* pour simplifier toutes les géométries lors de l'ingestion en utilisant l' Ramer-Douglas-Peuckeralgorithme et la tolérance donnée.
- Spécifiez `SIMPLIFY AUTO` sans tolérance pour simplifier uniquement les géométries supérieures à la taille maximale à l'aide de l' Ramer-Douglas-Peuckeralgorithme. Cette approche calcule la tolérance minimale suffisamment grande pour stocker l'objet dans la limite de taille maximale.
- Spécifiez `SIMPLIFY AUTO` *`max_tolerance`* pour simplifier uniquement les géométries supérieures à la taille maximale à l'aide de l' Ramer-Douglas-Peuckeralgorithme et de la tolérance calculée automatiquement. Cette approche permet de s'assurer que la tolérance ne dépasse pas la tolérance maximale.

Pour plus d'informations sur la taille maximale d'une valeur de données `GEOMETRY`, consultez Éléments à prendre en compte lors de l'utilisation de données spatiales dans Amazon Redshift.

Dans certains cas, la tolérance est suffisamment faible pour que l'enregistrement ne puisse pas passer en dessous de la taille maximale d'une valeur de données `GEOMETRY`. Dans ces cas, vous pouvez utiliser l'option `MAXERROR` de la commande COPY pour ignorer la totalité ou jusqu'à un certain nombre d'erreurs d'ingestion.

La commande COPY prend également en charge le chargement des shapefiles GZIP. Pour ce faire, spécifiez le paramètre COPY GZIP. Avec cette option, tous les composants du shapefile doivent être compressés indépendamment et partager le même suffixe de compression.

Si un fichier de description de projection (.prj) existe avec le fichier de formes, Redshift l'utilise pour déterminer l'ID du système de référence spatiale (SRID). Si le SRID est valide, ce SRID est attribué à la géométrie résultante. Si la valeur SRID associée à la géométrie d'entrée n'existe pas, la géométrie résultante a la valeur SRID zéro. Vous pouvez désactiver la détection automatique de l'ID du système de référence spatiale au niveau de la séance à l'aide de `SET read_srid_on_shapefile_ingestion` pour `OFF`.

Interrogez les vues système `SYS_SPATIAL_SIMPLIFY` ou `SVL_SPATIAL_SIMPLIFY` pour afficher les enregistrements qui ont été simplifiés ainsi que la tolérance calculée. Lorsque vous spécifiez `SIMPLIFY` *`tolerance`*, cette vue contient un enregistrement pour chaque opération COPY. Sinon, il contient un enregistrement pour chaque géométrie simplifiée. Pour plus d'informations, consultez SYS_SPATIAL_SIMPLIFY ou SVL_SPATIAL_SIMPLIFICATION.

Pour obtenir des exemples de chargement d'un shapefile, consultez Chargement d'un shapefile dans Amazon Redshift.

Terminologie des données spatiales Amazon Redshift

Les termes suivants sont utilisés pour décrire certaines fonctions spatiales Amazon Redshift.

Cadre de délimitation

Le cadre de délimitation d'une géométrie ou d'une géographie est défini comme le produit vectoriel (à travers les dimensions) des extensions des coordonnées de tous les points de la géométrie ou de la géographie. Pour les géométries bidimensionnelles, le cadre de délimitation est un rectangle qui inclut complètement tous les points de la géométrie. Par exemple, un cadre de délimitation du polygone `POLYGON((0 0,1 0,0 2,0 0))` est le rectangle défini par les points (0, 0) et (1, 2) comme étant le coin inférieur gauche et le coin supérieur droit. Amazon Redshift précalcule et stocke un cadre de délimitation dans une géométrie pour accélérer les prédicats géométriques et les jointures spatiales. Par exemple, si les cadres de délimitation de deux géométries ne se croisent pas, ces deux géométries ne peuvent pas se croiser et elles ne peuvent pas être dans le jeu de résultats d'une jointure spatiale à l'aide du prédicat ST_Intersects.

Vous pouvez utiliser des fonctions spatiales pour ajouter (Ajouter BBox), supprimer (Lâchez BBox) et déterminer le support (Supports BBox) pour un cadre de délimitation. Amazon Redshift prend en charge le précalcul des cadres de délimitation pour tous les sous-types de géométrie.

L'exemple suivant explique comment mettre à jour les géométries existantes d'une table afin de les stocker avec un cadre de délimitation. Si votre cluster est à la version 1.0.26809 ou ultérieure du cluster, toutes les nouvelles géométries sont créées avec un cadre de délimitation précalculé par défaut.

```
UPDATE my_table SET geom = AddBBox(geom) WHERE SupportsBBox(geom) = false;
```

Après avoir mis à jour les géométries existantes, nous vous recommandons d'exécuter la commande VACUUM sur la table mise à jour. Pour plus d'informations, consultez VACUUM.

Pour définir si les géométries sont encodées avec un cadre de délimitation au cours d'une session, consultez default_geometry_encoding.

Validité géométrique

Les algorithmes géométriques utilisés par Amazon Redshift supposent que la géométrie en entrée est une géométrie valide. Si une entrée dans un algorithme n'est pas valide, le résultat est indéfini.

La section suivante décrit les définitions de validité géométrique utilisées par Amazon Redshift pour chaque sous-type de géométrie.

Point

Un point est considéré comme valide si l'une des conditions suivantes est remplie :

- Le point est le point vide.
- Toutes les coordonnées des points sont des nombres finis à virgule flottante.

Un point peut être le point vide.

Linestring

Une linestring est considérée comme valide si l'une des conditions suivantes est remplie :

- La linestring est vide, c'est-à-dire qu'elle ne contient aucun point.
- Tous les points d'une linestring non vide ont des coordonnées qui sont des nombres finis à virgule flottante.
- Si elle n'est pas vide, la linestring doit être unidimensionnelle, c'est-à-dire qu'elle ne peut pas dégénérer en un point.

Une linestring ne peut pas contenir de points vides.

Une linestring peut avoir des points consécutifs en double.

Une linestring peut avoir des auto-intersections.

Polygon

Un polygone est considéré comme valide si l'une des conditions suivantes est remplie :

- Le polygone est vide, c'est-à-dire qu'il ne contient pas d'anneaux.
- S'il n'est pas vide, un polygone est valide si toutes les conditions suivantes sont remplies :
  - Tous les anneaux du polygone sont valides. Un anneau est considéré comme valide si toutes les conditions suivantes sont remplies :
    - Tous les points de l'anneau ont des coordonnées qui sont des nombres finis à virgule flottante.
    - L'anneau est fermé, c'est-à-dire que son premier point et son dernier point coïncident.
    - L'anneau n'a pas d'auto-intersections.
    - L'anneau est bidimensionnel.

- Les anneaux du polygone doivent avoir des orientations cohérentes. Autrement dit, si vous traversez un anneau, l'intérieur du polygone se trouve soit à votre droite, soit à votre gauche.
- Tous les anneaux intérieurs doivent se situer dans l'anneau extérieur du polygone.
- Tous les anneaux intérieurs doivent avoir la même orientation, et cette orientation doit être opposée à celle de l'anneau extérieur.
- Les anneaux intérieurs ne peuvent pas être imbriqués, c'est-à-dire qu'un anneau intérieur ne peut pas se trouver dans un autre anneau intérieur.
- Les anneaux intérieurs et extérieurs ne peuvent se croiser qu'à un nombre fini de points.
- L'intérieur du polygone doit être simplement connecté.

Un polygone ne peut pas contenir de points vides.

Multipoint

Un multipoint est considéré comme valide si l'une des conditions suivantes est remplie :

- Le multipoint est vide, c'est-à-dire qu'il ne contient aucun point.
- Un multipoint n'est pas vide et tous les points sont valides selon la définition de validité des points.

Un multipoint peut contenir un ou plusieurs points vides.

Un multipoint peut avoir des points en double.

Multilinestring

Une multilinestring est considérée comme valide si l'une des conditions suivantes est remplie :

- La multilinestring est vide, c'est-à-dire qu'il ne contient pas de linestrings.
- Toutes les lignes d'une multilinestring non vide sont valides selon la définition de validité de linestring.

Une multilinestring non vide qui se compose uniquement de lignes vides est considérée comme valide.

Une linestring vide dans une multilinestring n'affecte pas sa validité.

Une multilinestring peut avoir des linestrings avec des points consécutifs en double.

Une multilinestring peut avoir des auto-intersections.

Une multilinestring ne peut pas contenir de points vides.

Multipolygon

Un multipolygone est considéré comme valide si l'une des conditions suivantes est remplie :

- Le multipolygone ne contient aucun polygone (il est vide).
- Le multipolygone n'est pas vide et toutes les conditions suivantes sont remplies :
  - Tous les polygones du multipolygone sont valides.
  - Aucun polygone dans le multipolygone ne peut se croiser à un nombre infini de points. En particulier, cela implique que l'intérieur de deux polygones ne peut pas se croiser et qu'ils ne peuvent se toucher qu'à un nombre fini de points.

Un polygone vide dans un multipolygone n'invalide pas un multipolygone.

Un multipolygone ne peut pas contenir de points vides.

Collection de géométries

Une collection de géométries est considérée comme valide si l'une des conditions suivantes est remplie :

- La collection de géométries est vide, c'est-à-dire qu'elle ne contient aucune géométrie.
- Toutes les géométries d'une collection de géométries non vide sont valides.

Cette définition s'applique toujours, bien que de manière récursive, aux collections de géométries imbriquées.

Une collection de géométries peut contenir des points vides et des multipoints avec des points vides.

Simplicité géométrique

Les algorithmes géométriques utilisés par Amazon Redshift supposent que la géométrie en entrée est une géométrie valide. Si une entrée dans un algorithme n'est pas valide, la vérification de simplicité n'est pas définie. La section suivante décrit les définitions de simplicité géométrique utilisées par Amazon Redshift pour chaque sous-type de géométrie.

Point

Un point valide est considéré comme simple si l'une des conditions suivantes est remplie :

- Un point valide est toujours considéré comme simple.
- Un point vide est considéré comme simple.

Linestring

Une linestring valide est considérée comme simple si l'une des conditions suivantes est remplie :

- La linestring est vide.
- La linestring n'est pas vide et toutes les conditions suivantes sont remplies :
  - Il n'a pas de points consécutifs en double.
  - Il n'a pas d'auto-intersections, sauf peut-être pour son premier point et son dernier point, qui peuvent coïncider. En d'autres termes, la linestring ne peut pas avoir d'auto-intersections sauf aux points de limite.

Polygon

Un polygone valide est considéré comme simple s'il ne contient pas de points consécutifs en double.

Multipoint

Un multipoint valide est considéré comme simple si l'une des conditions suivantes est remplie :

- Le multipoint est vide, c'est-à-dire qu'il ne contient pas de point.
- Aucun point non vide du multipoint coïncide avec un autre.

Multilinestring

Une multilinestring valide est considérée comme simple si l'une des conditions suivantes est remplie :

- La multilinestring est vide.
- La multilinestring est non vide et toutes les conditions suivantes sont remplies :
  - Toutes ses linestrings sont simples.
  - Aucunes linestrings du multilinestring ne se croisent, sauf aux points qui sont des points de limite des deux linestrings.

Une multilinestring non vide constituée de linestrings vides uniquement est considérée comme vide.

Une linestring vide dans une multilinestring n'affecte pas sa simplicité.

Une linestring fermée dans une multilinestring ne peut se croiser avec aucune autre linestring dans la multilinestring.

Une multilinestring ne peut pas avoir de linestrings avec des points consécutifs en double.

Multipolygon

Un multipolygone valide est considéré comme simple s'il ne contient pas de points consécutifs en double.

Collection de géométries

Une collection de géométries valide est considérée comme simple si l'une des conditions suivantes est remplie :

- La collection de géométries est vide, c'est-à-dire qu'elle ne contient aucune géométrie.
- Toutes les géométries d'une collection de géométries non vide sont simples.

Cette définition s'applique toujours, bien que de manière récursive, aux collections de géométries imbriquées.

H3

H3 est un système de grille d'indexation géospatiale hiérarchique qui permet d'indexer les coordonnées spatiales jusqu'à une résolution au mètre carré. Les données indexées peuvent être regroupées dans des jeux de données disparates et agrégées à différents niveaux de précision. H3 permet une gamme d'algorithmes et d'optimisations basés sur la grille, y compris les voisins les plus proches, le chemin le plus court, le lissage des dégradés, etc. Les indices H3 font référence à des cellules qui peuvent être hexagonales ou pentagonales. L'espace est subdivisé hiérarchiquement selon une résolution. H3 prend en charge 16 résolutions de 0 à 15 incluses, `0` étant la résolution la plus grossière et `15` la plus fine.

Amazon Redshift fournit les fonctions spatiales H3 suivantes :

- H3_Boundary
- H3_Centre
- H3_ FromLongLat
- H3_ FromPoint
- H3_Polyfill

Éléments à prendre en compte lors de l'utilisation de données spatiales dans Amazon Redshift

Voici des éléments à prendre en compte lors de l'utilisation des données spatiales avec Amazon Redshift :

- La taille maximum d'un objet GEOMETRY ou GEOGRAPHY est de 1 048 447 octets.
- Amazon Redshift Spectrum ne prend pas en charge les données spatiales de façon native. Par conséquent, vous ne pouvez pas créer ni modifier de table externe avec une colonne GEOMETRY ou GEOGRAPHY.
- Les types de données pour les fonctions définies par l'utilisateur en Python (UDFs) ne prennent pas en charge le type de GEOGRAPHY données GEOMETRY or.
- Vous ne pouvez pas utiliser de colonne GEOMETRY ou GEOGRAPHY comme clé de tri ou clé de distribution d'une table Amazon Redshift.
- Vous ne pouvez pas utiliser de colonnes GEOMETRY ou GEOGRAPHY dans les clauses SQL ORDER BY, GROUP BY ou DISTINCT.
- Vous ne pouvez pas utiliser de colonnes GEOMETRY ou GEOGRAPHY dans de nombreuses fonctions SQL.
- Vous ne pouvez pas effectuer d'opération UNLOAD sur des colonnes GEOMETRY ou GEOGRAPHY dans chaque format. Vous pouvez effectuer l'opération UNLOAD sur les colonnes GEOMETRY ou GEOGRAPHY des fichiers texte ou CSV (valeurs séparées par des virgules). Si vous le faites, les données GEOMETRY ou GEOGRAPHY sont écrites au format EWKB hexadécimal. Si la taille des données EWKB est supérieure à 4 Mo, un avertissement est envoyé car les données ne pourront pas être chargées dans un table par la suite.
- L'encodage de compression pris en charge pour les données GEOMETRY ou GEOGRAPHY est RAW.
- Lorsque vous utilisez les pilotes JDBC ou ODBC, utilisez des mappages de type personnalisés. Dans ce cas, l'application cliente doit avoir des informations afin de savoir quels paramètres d'un objet ResultSet sont des objets GEOMETRY ou GEOGRAPHY. L'opération ResultSetMetadata renvoie le type VARCHAR.
- Pour copier une date géographique à partir d'un SHAPEFILE, effectuez d'abord une ingestion dans une colonne GEOMETRY, puis convertissez les objets en objets GEOGRAPHY.

Les fonctions non spatiales suivantes peuvent accepter une entrée de type GEOMETRY ou GEOGRAPHY ou des colonnes de type GEOMETRY ou GEOGRAPHY :

- Fonction d’agrégation COUNT
- Expressions conditionnelles COALESCE et NVL
- Expressions CASE
- L’encodage par défaut pour `GEOMETRY` et `GEOGRAPHY` est RAW. Pour plus d'informations, consultez encodages de compression.

Interrogation de données avec requête fédérée dans Amazon Redshift

Les requêtes fédérées dans Amazon Redshift vous permettent d'interroger et d'analyser des données dans des bases de données opérationnelles, des entrepôts des données et des lacs de données. Avec la fonction Requête fédérée, vous pouvez intégrer des requêtes à partir de données Amazon Redshift en direct dans des bases de données externes avec des requêtes dans vos environnements Amazon Redshift et Amazon S3. Les requêtes fédérées peuvent fonctionner avec des bases de données externes dans Amazon RDS for PostgreSQL, Amazon Aurora Édition compatible avec PostgreSQL, Amazon RDS for MySQL et Amazon Aurora Édition compatible avec PostgreSQL.

Vous pouvez utiliser les requêtes fédérées pour incorporer des données en direct dans le cadre de votre Business Intelligence (BI) et de vos applications de génération de rapports. Par exemple, pour faciliter l'ingestion de données dans Amazon Redshift, vous pouvez utiliser des requêtes fédérées pour effectuer les opérations suivantes :

- Interroger directement les bases de données opérationnelles.
- Appliquer rapidement les transformations.
- Charger les données dans les tables cible sans avoir besoin de pipelines complexes d'extraction, de transformation, de chargement (ETL).

Pour réduire le déplacement des données sur le réseau et améliorer les performances, Amazon Redshift distribue une partie du calcul des requêtes fédérées directement dans les bases de données opérationnelles distantes. Amazon Redshift utilise également sa capacité de traitement parallèle pour prendre en charge l'exécution de ces requêtes, selon les besoins.

Lors de l'exécution de requêtes fédérées, Amazon Redshift établit d'abord une connexion client à l'instance de base de données du cluster de base de données RDS ou Aurora à partir du nœud principal pour récupérer les métadonnées des tables. À partir d'un nœud de calcul, Amazon Redshift émet des sous-requêtes avec un prédicat poussé vers le bas et récupère les lignes de résultat. Amazon Redshift distribue ensuite les lignes de résultat entre les nœuds de calcul pour un traitement ultérieur.

Les détails sur les requêtes envoyées à la base de données Amazon Aurora PostgreSQL ou à la base de données Amazon RDS for PostgreSQL sont journalisés dans la vue système SVL_FEDERATED_QUERY.

Rubriques

- Commencer à utiliser les requêtes fédérées vers PostgreSQL
- Commencer à utiliser des requêtes fédérées dans PostgreSQL avec AWS CloudFormation
- Commencer à utiliser des requêtes fédérées vers MySQL
- Création d'un secret et d'un rôle IAM pour utiliser des requêtes fédérées
- Exemples d'utilisation d'une requête fédérée
- Différences de type de données entre Amazon Redshift et les bases de données PostgreSQL et MySQL prises en charge
- Éléments à prendre en compte lors de l'accès aux données fédérées avec Amazon Redshift

Commencer à utiliser les requêtes fédérées vers PostgreSQL

Pour créer une requête fédérée, suivez l'approche générale suivante :

1. Configurez la connectivité entre votre cluster Amazon Redshift et votre instance de base de données PostgreSQL Amazon RDS ou Aurora.

   Pour ce faire, assurez-vous que votre instance de base de données RDS PostgreSQL ou Aurora PostgreSQL peut accepter les connexions de votre cluster Amazon Redshift. Nous recommandons que votre cluster Amazon Redshift et instance PostgreSQL d'Amazon RDS ou Aurora soient dans le même cloud privé virtuel (VPC) et le même groupe de sous-réseaux. De cette façon, vous pouvez ajouter le groupe de sécurité du cluster Amazon Redshift aux règles entrantes du groupe de sécurité pour votre instance de base de données RDS ou Aurora PostgreSQL.

   Vous pouvez également configurer l'appairage de VPC ou d'autres réseaux qui permettent à Amazon Redshift d'établir des connexions à votre instance RDS ou Aurora PostgreSQL. Pour plus d'informations sur la mise en réseau de VPC, consultez les informations suivantes :

   - Qu'est-ce que l'appairage de VPC ? dans le Guide d'appairage de VPC Amazon
   - Utilisation d'une instance de base de données dans un VPC dans le Guide de l'utilisateur Amazon RDS

   > **Note**
   >
   > Dans certains cas, vous devez activer le routage VPC amélioré : par exemple, si votre cluster Amazon Redshift se trouve dans un VPC différent de celui de votre instance RDS

ou Aurora PostgreSQL, ou s'ils se trouvent dans le même VPC et que vos itinéraires l'exigent. Sinon, vous risquez de recevoir des erreurs de délai d'expiration lorsque vous exécutez une requête fédérée.

2. Configurez des secrets AWS Secrets Manager pour vos bases de données RDS PostgreSQL et Aurora PostgreSQL. Référencez ensuite les secrets dans les politiques d'accès et les rôles AWS Identity and Access Management (IAM). Pour de plus amples informations, veuillez consulter Création d'un secret et d'un rôle IAM pour utiliser des requêtes fédérées.

   > **Note**
   >
   > Si votre cluster utilise un routage VPC amélioré, vous devrez peut-être configurer un point de terminaison de VPC d'interface pour AWS Secrets Manager. Cela est nécessaire lorsque le VPC et le sous-réseau de votre cluster Amazon Redshift n'ont pas accès au point de terminaison public. AWS Secrets Manager Lorsque vous utilisez un point de terminaison d'interface VPC, la communication entre le cluster Amazon Redshift de votre VPC est acheminée de manière privée depuis votre VPC AWS Secrets Manager vers l'interface de point de terminaison. Pour plus d'informations, consultez Création d'un point de terminaison d'interface dans le Guide de l'utilisateur Amazon VPC.

3. Appliquez le rôle IAM que vous avez créé précédemment au cluster Amazon Redshift. Pour de plus amples informations, veuillez consulter Création d'un secret et d'un rôle IAM pour utiliser des requêtes fédérées.
4. Connectez-vous à votre RDS PostgreSQL et à vos bases de données Aurora PostgreSQL à l'aide d'un schéma externe. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL SCHEMA. Pour obtenir des exemples sur l'utilisation de la requête fédérée, consultez Exemples d'utilisation d'une requête fédérée.
5. Exécutez vos requêtes SQL référençant le schéma externe qui fait référence à votre RDS PostgreSQL et à vos bases de données Aurora PostgreSQL.

Commencer à utiliser des requêtes fédérées dans PostgreSQL avec AWS CloudFormation

Vous pouvez utiliser des requêtes fédérées pour interroger des bases de données opérationnelles. Dans ce guide de démarrage, vous pouvez automatiser la configuration en utilisant un exemple de AWS CloudFormation pile pour activer une requête fédérée depuis un cluster Amazon Redshift vers

une base de données sans serveur Aurora PostgreSQL. Vous pouvez être rapidement opérationnel sans avoir à exécuter des instructions SQL pour allouer vos ressources.

La pile crée un schéma externe, faisant référence à votre instance Aurora PostgreSQL, qui inclut des tables contenant des exemples de données. Vous pouvez interroger des tables dans le schéma externe à partir de votre cluster Redshift.

Si vous souhaitez plutôt commencer à utiliser des requêtes fédérées en exécutant des instructions SQL pour configurer un schéma externe, sans utiliser CloudFormation, consultezCommencer à utiliser les requêtes fédérées vers PostgreSQL.

Avant d'exécuter la CloudFormation pile pour les requêtes fédérées, assurez-vous que vous disposez d'une base de données sans serveur Amazon Aurora PostgreSQL Edition avec l'API Data activée. Vous pouvez activer l'API de données dans les propriétés de la base de données. Si vous ne trouvez pas le paramètre, vérifiez à nouveau que vous exécutez une instance sans serveur d'Aurora PostgreSQL. Assurez-vous également que vous disposez d'un cluster Amazon Redshift qui utilise des RA3 nœuds. Nous recommandons que votre cluster Redshift et votre instance PostgreSQL d'Aurora sans serveur ou Aurora soient dans le même cloud privé virtuel (VPC) et le même groupe de sous-réseaux. De cette façon, vous pouvez ajouter le groupe de sécurité du cluster Amazon Redshift aux règles entrantes du groupe de sécurité pour votre instance de base de données Aurora PostgreSQL.

Pour plus d'informations sur la mise en place d'un cluster Amazon Redshift, consultez Commencer avec les entrepôts de données provisionnés Amazon Redshift. Pour plus d'informations sur la configuration des ressources avec CloudFormation, voir Qu'est-ce que c'est AWS CloudFormation ? . Pour plus d'informations sur la configuration d'une base de données de cluster de base de données Aurora, voir Création d'un cluster de base de données Aurora Serverless v1.

Lancement d'une CloudFormation pile pour les requêtes fédérées Redshift

Utilisez la procédure suivante pour lancer votre CloudFormation stack pour Amazon Redshift afin d'activer les requêtes fédérées. Avant de le faire, assurez-vous que votre cluster Amazon Redshift et votre instance Aurora PostgreSQL sans serveur sont configurés.

Pour lancer votre CloudFormation stack pour les requêtes fédérées

1. Cliquez sur Lancer la pile CFN ici pour lancer le CloudFormation service dans le AWS Management Console.

   Si vous y êtes invité, connectez-vous.

Le processus de création de la pile démarre en faisant référence à un fichier CloudFormation modèle, qui est stocké dans Amazon S3. Un CloudFormation modèle est un fichier texte au format JSON qui déclare AWS les ressources qui constituent une pile.

2. Choisissez Next (Suivant) pour saisir les détails de la pile.
3. Sous Parameters (Paramètres), pour le cluster, saisissez les éléments suivants :

   - Le nom du cluster Amazon Redshift, par exemple **`ra3-consumer-cluster`**
   - Un nom de base de données spécifique, par exemple **`dev`**
   - Le nom d'un utilisateur de base de données, par exemple **`consumeruser`**

   Entrez également les paramètres de la base de données du cluster de base de données Aurora, notamment l'utilisateur, le nom de la base de données, le port et le point de terminaison. Nous vous recommandons d'utiliser un cluster de test et une base de données sans serveur de test, car la pile crée plusieurs objets de base de données.

   Choisissez Suivant.

   Les options de pile apparaissent.
4. Choisissez Next (Suivant) pour accepter les paramètres par défaut.
5. Sous Fonctionnalités, choisissez Je reconnais que cela AWS CloudFormation pourrait créer des ressources IAM.
6. Sélectionnez Créer la pile.

Choisissez Create stack. CloudFormation approvisionne les ressources du modèle, ce qui prend environ 10 minutes, et crée un schéma externe.

Si une erreur se produit pendant la création de la pile, procédez comme suit :

- Consultez l'onglet CloudFormation Événements pour obtenir des informations qui peuvent vous aider à résoudre l'erreur.
- Assurez-vous d'avoir saisi le nom, le nom de base de données et le nom d'utilisateur de la base de données corrects pour le cluster Redshift. Vérifiez également les paramètres de l'instance Aurora PostgreSQL.
- Assurez-vous que votre cluster possède RA3 des nœuds.

- Assurez-vous que votre base de données et votre cluster Redshift se trouvent dans le même sous-réseau et le même groupe de sécurité.

Interrogation de données à partir du schéma externe

Pour utiliser la procédure ci-dessous, vérifiez que vous disposez des autorisations requises pour exécuter des requêtes sur le cluster et sur la base de données décrits.

Pour interroger une base de données externe avec une requête fédérée

1. Connectez-vous à la base de données Redshift que vous avez entrée lorsque vous avez créé la pile, à l'aide d'un outil client tel que l'éditeur de requêtes Redshift.
2. Interrogez le schéma externe créé par la pile.

   ```
   select * from svv_external_schemas;
   ```

   La vue SVV_EXTERNAL_SCHEMAS renvoie des informations sur les schémas externes disponibles. Dans ce cas, le schéma externe créé par la pile est renvoyé, `myfederated_schema`. D'autres schémas externes peuvent également être renvoyés, si vous en avez configurés. La vue renvoie également la base de données associée du schéma. La base de données est la base de données du cluster de base de données Aurora que vous avez saisie lors de la création de la pile. La pile ajoute une table à la base de données du cluster de base de données Aurora, appelée`category`, et une autre table appelée`sales`.
3. Exécutez des requêtes SQL sur des tables dans le schéma externe qui fait référence à votre base de données Aurora PostgreSQL. L'exemple suivant montre une requête.

   ```
   SELECT count(*) FROM myfederated_schema.category;
   ```

   La table `category` renvoie plusieurs enregistrements. Vous pouvez également renvoyer des enregistrements à partir de la table `sales`.

   ```
   SELECT count(*) FROM myfederated_schema.sales;
   ```

   Pour obtenir plus d'exemples, consultez Exemples d'utilisation d'une requête fédérée.

Commencer à utiliser des requêtes fédérées vers MySQL

Pour créer une requête fédérée vers des bases de données MySQL, suivez l'approche générale suivante :

1. Configurez la connectivité entre votre cluster Amazon Redshift et votre instance de base de données MySQL Amazon RDS ou Aurora.

   Pour ce faire, assurez-vous que votre instance de base de données RDS MySQL ou Aurora MySQL peut accepter les connexions de votre cluster Amazon Redshift. Nous recommandons que votre cluster Amazon Redshift et votre instance MySQL Amazon RDS ou Aurora MySQL soient dans le même cloud privé virtuel (VPC) et le même groupe de sous-réseaux. De cette façon, vous pouvez ajouter le groupe de sécurité du cluster Amazon Redshift aux règles entrantes du groupe de sécurité pour votre instance de base de données RDS ou Aurora MySQL.

   Vous pouvez également configurer l'appairage de VPC ou d'autres réseaux qui permettent à Amazon Redshift d'établir des connexions à votre instance RDS ou Aurora MySQL. Pour plus d'informations sur la mise en réseau de VPC, consultez les informations suivantes :

   - Qu'est-ce que l'appairage de VPC ? dans le Guide d'appairage de VPC Amazon
   - Utilisation d'une instance de base de données dans un VPC dans le Guide de l'utilisateur Amazon RDS

   > **Note**
   >
   > Si votre cluster Amazon Redshift se trouve dans un VPC différent de celui de votre instance RDS ou Aurora MySQL, activez le routage VPC amélioré. Sinon, vous risquez de recevoir des erreurs de délai d'expiration lorsque vous exécutez une requête fédérée.

2. Configurez des secrets AWS Secrets Manager pour vos bases de données RDS MySQL et Aurora MySQL. Référencez ensuite les secrets dans les politiques d'accès et les rôles AWS Identity and Access Management (IAM). Pour de plus amples informations, veuillez consulter Création d'un secret et d'un rôle IAM pour utiliser des requêtes fédérées.

   > **Note**
   >
   > Si votre cluster utilise un routage VPC amélioré, vous devrez peut-être configurer un point de terminaison de VPC d'interface pour AWS Secrets Manager. Cela est nécessaire lorsque le VPC et le sous-réseau de votre cluster Amazon Redshift n'ont pas accès au

point de terminaison public. AWS Secrets Manager Lorsque vous utilisez un point de terminaison de VPC d'interface, la communication entre le cluster Amazon Redshift de votre VPC et AWS Secrets Manager est acheminée en privé depuis votre VPC vers l'interface de point de terminaison. Pour plus d'informations, consultez Création d'un point de terminaison d'interface dans le Guide de l'utilisateur Amazon VPC.

3. Appliquez le rôle IAM que vous avez créé précédemment au cluster Amazon Redshift. Pour de plus amples informations, veuillez consulter Création d'un secret et d'un rôle IAM pour utiliser des requêtes fédérées.
4. Connectez-vous à votre RDS MySQL et à vos bases de données Aurora MySQL à l'aide d'un schéma externe. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL SCHEMA. Pour obtenir des exemples sur l'utilisation de la requête fédérée, consultez Exemple d'utilisation d'une requête fédérée avec MySQL.
5. Exécutez vos requêtes SQL référençant le schéma externe qui fait référence à votre RDS MySQL et à vos bases de données Aurora MySQL.

Création d'un secret et d'un rôle IAM pour utiliser des requêtes fédérées

Les étapes suivantes montrent comment créer un secret et un rôle IAM à utiliser avec une requête fédérée.

Prérequis

Assurez-vous de disposer des prérequis suivants pour créer un secret et un rôle IAM à utiliser avec une requête fédérée :

- Une instance de base de données RDS PostgreSQL, Aurora PostgreSQL, RDS MySQL ou Aurora MySQL avec authentification par nom d'utilisateur et mot de passe.
- Un cluster Amazon Redshift avec une version de maintenance de cluster prenant en charge les requêtes fédérées.

Pour créer un secret (nom d'utilisateur et mot de passe) avec AWS Secrets Manager

1. Connectez-vous à la console Secrets Manager avec le compte propriétaire de votre instance de cluster de base de données RDS ou Aurora.

2. Choisissez Store a new secret (Stocker un nouveau secret).
3. Choisissez la vignette Informations d'identification pour une base de données RDS . Pour Nom d'utilisateur et Mot de passe, entrez des valeurs pour votre instance. Confirmez ou choisissez une valeur pour Encryption key (Clé de chiffrement). Choisissez ensuite la base de données RDS à laquelle votre secret accédera.

> **Note**
>
> Nous vous recommandons d'utiliser la clé de chiffrement par défaut (`DefaultEncryptionKey`). Si vous utilisez une clé de chiffrement personnalisée, le rôle IAM utilisé pour accéder au secret doit être ajouté en tant qu'utilisateur clé.

4. Entrez un nom pour le secret, poursuivez les étapes de création avec les options par défaut, puis choisissez Stocker.
5. Affichez votre secret et notez la valeur de l'ARN du secret que vous avez créé pour identifier le secret.

Pour créer une politique de sécurité à l'aide du secret

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Créez une politique avec JSON similaire à ce qui suit.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-west-2:123456789012:secret:my-
rds-secret-VNenFy"
        },
        {
```

```
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Vous avez besoin des actions de liste et de lecture pour récupérer le secret. Nous vous recommandons de limiter la ressource au secret spécifique que vous avez créé. Pour ce faire, utilisez l'ARN (Amazon Resource Name) du secret afin de limiter la ressource. Vous pouvez également spécifier les autorisations et les ressources à l'aide de l'éditeur visuel de la console IAM.

3. Donnez un nom à la politique et terminez sa création.
4. Accédez à IAM roles (Rôles IAM).
5. Créez un rôle IAM pour Redshift - Customizable (Redshift - Personnalisable).
6. Attachez la politique IAM que vous venez de créer à un rôle IAM existant ou créez un nouveau rôle IAM et attachez-lui la stratégie.
7. Dans l'onglet Relations d'approbation de votre rôle IAM, confirmez que le rôle contient l'entité d'approbation `redshift.amazonaws.com`.
8. Notez l'ARN de rôle que vous avez créé. Cet ARN a accès au secret.

Pour attacher le rôle IAM à votre cluster Amazon Redshift

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters. Les clusters associés à votre compte dans la AWS région actuelle sont répertoriés.
3. Choisissez le nom du cluster dans la liste pour afficher plus de détails sur un cluster.
4. Sous Actions, choisissez Gérer les rôles IAM. La page Gérer les rôles IAM s'affiche.
5. Ajoutez votre rôle IAM au cluster.

Exemples d'utilisation d'une requête fédérée

Les exemples suivants illustrent l'exécution d'une requête fédérée. Exécutez l'instruction SQL à l'aide de votre client SQL connecté à la base de données Amazon Redshift.

Exemple d'utilisation d'une requête fédérée avec PostgreSQL

L'exemple suivant montre comment configurer une requête fédérée qui référence une base de données Amazon Redshift, une base de données Aurora PostgreSQL et Amazon S3. Cet exemple illustre le fonctionnement des requêtes fédérées. Pour l'exécuter sur votre propre environnement, modifiez-la afin qu'elle s'adapte à votre environnement. Pour connaître les conditions préalables à cette fin, consultez Commencer à utiliser les requêtes fédérées vers PostgreSQL.

Créez un schéma externe qui référence une base de données Aurora PostgreSQL.

```
CREATE EXTERNAL SCHEMA apg
FROM POSTGRES
DATABASE 'database-1' SCHEMA 'myschema'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Créez un autre schéma externe qui référence Amazon S3, qui utilise Amazon Redshift Spectrum. En outre, définissez l'autorisation d'utiliser le schéma sur `public`.

```
CREATE EXTERNAL SCHEMA s3
FROM DATA CATALOG
DATABASE 'default' REGION 'us-west-2'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-S3';

GRANT USAGE ON SCHEMA s3 TO public;
```

Affiche le nombre de lignes dans la table Amazon Redshift.

```
SELECT count(*) FROM public.lineitem;

  count
----------
25075099
```

Affiche le nombre de lignes dans la table Aurora PostgreSQL.

```
SELECT count(*) FROM apg.lineitem;

count
-------
11760
```

Affiche le nombre de lignes dans Amazon S3.

```
SELECT count(*) FROM s3.lineitem_1t_part;

   count
------------
6144008876
```

Créez une vue des tables depuis Amazon Redshift, Aurora PostgreSQL et Amazon S3. Cette vue est utilisée pour exécuter votre requête fédérée.

```
CREATE VIEW lineitem_all AS
  SELECT
 l_orderkey,l_partkey,l_suppkey,l_linenumber,l_quantity,l_extendedprice,l_discount,l_tax,l_retu
        l_shipdate::date,l_commitdate::date,l_receiptdate::date,
 l_shipinstruct ,l_shipmode,l_comment
  FROM s3.lineitem_1t_part
  UNION ALL SELECT * FROM public.lineitem
  UNION ALL SELECT * FROM apg.lineitem
     with no schema binding;
```

Affiche le nombre de lignes dans la vue `lineitem_all` avec un prédicat pour limiter les résultats.

```
SELECT count(*) from lineitem_all WHERE l_quantity = 10;

   count
-----------
123373836
```

Découvrez le nombre de ventes pour un article en janvier de chaque année.

```
SELECT extract(year from l_shipdate) as year,
```

```
       extract(month from l_shipdate) as month,
       count(*) as orders
FROM lineitem_all
WHERE extract(month from l_shipdate) = 1
AND l_quantity < 2
GROUP BY 1,2
ORDER BY 1,2;

 year | month | orders
------+-------+---------
 1992 |     1 |  196019
 1993 |     1 | 1582034
 1994 |     1 | 1583181
 1995 |     1 | 1583919
 1996 |     1 | 1583622
 1997 |     1 | 1586541
 1998 |     1 | 1583198
 2016 |     1 |   15542
 2017 |     1 |   15414
 2018 |     1 |   15527
 2019 |     1 |     151
```

Exemple d’utilisation d’un nom en casse mixte

Pour interroger une base de données distante PostgreSQL prise en charge et dont le nom de base de données, de schéma, de table ou de colonne comporte une casse mixte, définissez `enable_case_sensitive_identifier` sur `true`. Pour plus d’informations sur le paramètre de cette session, consultez enable_case_sensitive_identifier.

```
SET enable_case_sensitive_identifier TO TRUE;
```

En général, les noms de base de données et de schéma sont en minuscules. L’exemple suivant montre comment vous pouvez vous connecter à une base de données distante PostgreSQL prise en charge qui a des noms en minuscules pour la base de données et le schéma et des noms en casse mixte pour la table et la colonne.

Créez un schéma externe qui référence une base de données Aurora PostgreSQL ayant un nom de base de données en minuscules (`dblower`) et un nom de schéma en minuscules (`schemalower`).

```
CREATE EXTERNAL SCHEMA apg_lower
FROM POSTGRES
```

```
DATABASE 'dblower' SCHEMA 'schemalower'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Dans la session où la requête s'exécute, définissez `enable_case_sensitive_identifier` sur `true`.

```
SET enable_case_sensitive_identifier TO TRUE;
```

Exécutez une requête fédérée pour sélectionner toutes les données de la base de données PostgreSQL. La table (`MixedCaseTab`) et la colonne (`MixedCaseName`) ont des noms en casse mixte. Le résultat est une ligne (`Harry`).

```
select * from apg_lower."MixedCaseTab";
```

```
 MixedCaseName
-------
 Harry
```

L'exemple suivant montre comment vous pouvez vous connecter à une base de données distante PostgreSQL prise en charge, dont le nom de la base de données, du schéma, de la table et de la colonne sont en casse mixte.

Définissez `enable_case_sensitive_identifier` sur `true` avant de créer le schéma externe. Si `enable_case_sensitive_identifier` n'est pas défini sur `true` avant de créer le schéma externe, une erreur de non-existence de la base de données (database does not exist) se produit.

Créez un schéma externe qui fait référence à une base de données Aurora PostgreSQL dont le nom de base de données (`UpperDB`) et de schéma (`UpperSchema`) est en casse mixte.

```
CREATE EXTERNAL SCHEMA apg_upper
FROM POSTGRES
DATABASE 'UpperDB' SCHEMA 'UpperSchema'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Exécutez une requête fédérée pour sélectionner toutes les données de la base de données PostgreSQL. La table (`MixedCaseTab`) et la colonne (`MixedCaseName`) ont des noms en casse mixte. Le résultat est une ligne (`Harry`).

```
select * from apg_upper."MixedCaseTab";
```

```
 MixedCaseName
-------
 Harry
```

Exemple d'utilisation d'une requête fédérée avec MySQL

L'exemple suivant montre comment configurer une requête fédérée qui référence une base de données Aurora MySQL. Cet exemple illustre le fonctionnement d'une requête fédérée. Pour l'exécuter sur votre propre environnement, modifiez-la afin qu'elle s'adapte à votre environnement. Pour connaître les conditions préalables à cette fin, consultez Commencer à utiliser des requêtes fédérées vers MySQL.

Cet exemple dépend des prérequis suivants :

- Un secret qui a été configuré dans Secrets Manager pour la base de données Aurora MySQL. Ce secret est référencé dans les stratégies d'accès et les rôles IAM. Pour de plus amples informations, veuillez consulter Création d'un secret et d'un rôle IAM pour utiliser des requêtes fédérées.
- Groupe de sécurité qui est configuré pour lier Amazon Redshift et Aurora MySQL.

Créez un schéma externe qui référence une base de données Aurora MySQL.

```
CREATE EXTERNAL SCHEMA amysql
FROM MYSQL
DATABASE 'functional'
URI 'endpoint to remote hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-SecretsManager-RO'
SECRET_ARN 'arn:aws:secretsmanager:us-west-2:123456789012:secret:federation/test/
dataplane-apg-creds-YbVKQw';
```

Exécutez un exemple de sélection SQL de la table Aurora MySQL pour afficher une ligne de la table des employés dans Aurora MySQL.

```
SELECT level FROM amysql.employees LIMIT 1;

 level
-------
     8
```

Différences de type de données entre Amazon Redshift et les bases de données PostgreSQL et MySQL prises en charge

Le tableau suivant montre le mappage d'un type de données Amazon Redshift à un type de données Amazon RDS PostgreSQL ou Aurora PostgreSQL correspondant.

Type de données Amazon Redshift	Type de données RDS PostgreSQL ou Aurora PostgreSQL	Description
SMALLINT	SMALLINT	Entier signé sur deux octets
INTEGER	INTEGER	Entier signé sur quatre octets
BIGINT	BIGINT	Entier signé sur huit octets
DECIMAL	DECIMAL	Valeur numérique exacte avec précision sélectionnable
REAL	REAL	Nombre à virgule flottante simple précision
DOUBLE PRECISION	DOUBLE PRECISION	Nombre à virgule flottante de double précision

Type de données Amazon Redshift	Type de données RDS PostgreSQL ou Aurora PostgreSQL	Description
BOOLEAN	BOOLEAN	Booléen logique (true/false)
CHAR	CHAR	Chaîne de caractères de longueur fixe
VARCHAR	VARCHAR	Chaîne de caractères de longueur variable avec une limite définie par l'utilisateur
DATE	DATE	Date calendaire (année, mois, jour)
TIMESTAMP	TIMESTAMP	Date et heure (sans fuseau horaire)
TIMESTAMPTZ	TIMESTAMPTZ	Date et heure (avec fuseau horaire)
GEOMETRY	PostGIS GEOMETRY	Données spatiales

Les types de données RDS PostgreSQL et Aurora PostgreSQL suivants sont convertis en VARCHAR (64K) dans Amazon Redshift :

- JSON, JSONB
- Arrays (tableaux)
- BIT, BIT VARYING
- BYTEA
- Types composites
- Type de date et d'heure INTERVAL, TIME, TIME WITH TIMEZONE
- Types énumérés
- Types monétaires

- Types d’adresse réseau
- Types numériques SERIAL, BIGSERIAL, SMALLSERIAL et MONEY
- Types d’identifiant d’objet
- pg_lsn type
- Pseudotypes
- Types de plage
- Types de recherche de texte
- TXID_SNAPSHOT
- UUID
- Type XML

Le tableau suivant montre le mappage d’un type de données Amazon Redshift à un type de données Amazon RDS MySQL ou Aurora MySQL correspondant.

Type de données Amazon Redshift	Type de données RDS MySQL ou Aurora MySQL	Description
BOOLEAN	TINYINT(1)	Booléen logique (true ou false)
SMALLINT	TINYINT(UNSIGNED)	Entier signé sur deux octets
SMALLINT	SMALLINT	Entier signé sur deux octets
INTEGER	SMALLINT UNSIGNED	Entier signé sur quatre octets
INTEGER	MEDIUMINT (UNSIGNED)	Entier signé sur quatre octets
INTEGER	INT	Entier signé sur quatre octets

Type de données Amazon Redshift	Type de données RDS MySQL ou Aurora MySQL	Description
BIGINT	INT UNSIGNED	Entier signé sur huit octets
BIGINT	BIGINT	Entier signé sur huit octets
DECIMAL	BIGINT UNSIGNED	Valeur numérique exacte avec précision sélectionnable
DECIMAL	DECIMAL(M,D)	Valeur numérique exacte avec précision sélectionnable
REAL	FLOAT	Nombre à virgule flottante simple précision
DOUBLE PRECISION	DOUBLE	Nombre à virgule flottante de double précision
CHAR	CHAR	Chaîne de caractères de longueur fixe
VARCHAR	VARCHAR	Chaîne de caractères de longueur variable avec une limite définie par l'utilisateur
DATE	DATE	Date calendaire (année, mois, jour)
TIME	TIME	TIME WITHOUT TIME ZONE

Type de données Amazon Redshift	Type de données RDS MySQL ou Aurora MySQL	Description
TIMESTAMP	TIMESTAMP	Date et heure (sans fuseau horaire)
TIMESTAMP	DATETIME	TIME WITHOUT TIME ZONE
VARCHAR(4)	YEAR	Caractère de longueur variable représentant l'année

Une erreur se produit lorsque les données TIME sont hors de portée (00:00:00 – 24:00:00).

Les types de données RDS MySQL et Aurora MySQL suivants sont convertis en VARCHAR (64K) dans Amazon Redshift :

- BIT
- BINAIRE
- VARBINARY
- TINYBLOB, BLOB, MEDIUMBLOB, LONGBLOB
- TINYTEXT, TEXT, MEDIUMTEXT, LONGTEXT
- ENUM
- SET
- SPATIAL

Éléments à prendre en compte lors de l'accès aux données fédérées avec Amazon Redshift

Certaines fonctions Amazon Redshift ne prennent pas en charge l'accès aux données fédérées. Vous trouverez ci-dessous les limitations et considérations connexes.

Voici des limites et considérations à prendre en compte lors de l'utilisation de requêtes fédérées avec Amazon Redshift :

- Les requêtes fédérées prennent en charge l’accès en lecture aux sources de données externes. Vous ne pouvez pas écrire ou créer des objets de base de données dans la source de données externe.
- Dans certains cas, vous pouvez accéder à une base de données de cluster Amazon RDS ou Aurora DB dans une AWS région différente de celle d'Amazon Redshift. Dans ces cas, vous devez généralement payer des frais de latence réseau et de facturation pour le transfert de données entre AWS les régions. Nous vous recommandons d'utiliser une base de données globale Aurora avec un point de terminaison local dans la même AWS région que votre cluster Amazon Redshift. Les bases de données globales Aurora utilisent une infrastructure dédiée pour la réplication basée sur le stockage entre deux régions AWS quelles qu’elles soient, avec une latence typique de moins d’une seconde.
- Tenez compte du coût d'accès au cluster de base de données Amazon RDS ou Aurora. Par exemple, lorsque vous utilisez cette fonctionnalité pour accéder au cluster de base de données Aurora, les frais de cluster de base de données Aurora sont basés sur les IOPS.
- Les requêtes fédérées ne permettent pas d'accéder à Amazon Redshift depuis un cluster de base de données RDS ou Aurora.
- Les requêtes fédérées ne sont disponibles que dans AWS les régions où Amazon Redshift et Amazon RDS ou le cluster de base de données Aurora sont disponibles.
- Pour l’instant, les requêtes fédérées ne prennent pas en charge `ALTER SCHEMA`. Pour modifier un schéma, utilisez `DROP` et puis `CREATE EXTERNAL SCHEMA`.
- Les requêtes fédérées ne fonctionnent pas avec la mise à l’échelle simultanée.
- Pour l’instant, les requêtes fédérées ne prennent actuellement pas en charge l’accès via un wrapper de données distantes PostgreSQL (PostgreSQL Foreign Data Wrapper).
- Les requêtes fédérées vers RDS MySQL ou Aurora MySQL prennent en charge l’isolement des transactions au niveau READ COMMITED.
- S’il n’est pas spécifié, Amazon Redshift se connecte à RDS for MySQL ou à Aurora MySQL sur le port 3306. Vérifiez le numéro de port MySQL avant de créer un schéma externe pour MySQL.
- S’il n’est pas spécifié, Amazon Redshift se connecte à RDS PostgreSQL ou à Aurora PostgreSQL sur le port 5432. Vérifiez le numéro de port PostgreSQL avant de créer un schéma externe pour PostgreSQL.
- Lors de la récupération de types de données TIMESTAMP et DATE à partir de MySQL, les valeurs zéro sont traitées comme NULL.

- Si un point de terminaison du lecteur de base de données du cluster de base de données Aurora est utilisé, une erreur « snapshot non valide » peut se produire. Cette situation peut être évitée par l'une des méthodes suivantes :
  - Utilisez un point de terminaison d'instance de cluster de base de données Aurora spécifique (au lieu d'utiliser le point de terminaison de cluster de base de données Aurora). Cette méthode utilise l'isolation des transactions REPEATABLE READ pour les résultats de la base de données PostgreSQL.
  - Utilisez un point de terminaison de lecteur de cluster de base de données Aurora et `pg_federation_repeatable_read` définissez-le sur false pour la session. Cette méthode utilise l'isolation des transactions READ COMMITTED pour les résultats de la base de données PostgreSQL. Pour plus d'informations sur les points de terminaison du lecteur de cluster de base de données Aurora, consultez la section Types de points de terminaison du cluster de base de données Aurora dans le guide de l'utilisateur Amazon Aurora. Pour de plus amples informations sur `pg_federation_repeatable_read`, consultez pg_federation_repeatable_read.

Les considérations suivantes concernent les transactions lors de l'utilisation de requêtes fédérées vers des bases de données PostgreSQL :

- Si une requête est constituée de tables fédérées, le nœud de ligne démarre une transaction READ ONLY REPEATABLE READ sur la base de données distante. Cette transaction reste pour la durée de la transaction Amazon Redshift.
- Le nœud leader crée un instantané de la base de données distante en appelant `pg_export_snapshot` et établit un verrou en lecture sur les tables affectées.
- Un nœud de calcul démarre une transaction et utilise l'instantané créé au niveau du nœud de référence pour émettre des requêtes vers la base de données distante.

Versions prises en charge des bases de données fédérées

Un schéma externe Amazon Redshift référence une base de données dans un RDS PostgreSQL ou Aurora PostgreSQL externe. Dans ce cas, les limites suivantes s'appliquent :

- Lors de la création d'un schéma externe référençant un cluster de base de données Aurora, la base de données Aurora PostgreSQL doit être de version 9.6 ou ultérieure.
- Lors de la création d'un schéma externe référençant Amazon RDS, la base de données Amazon RDS PostgreSQL doit être à la version 9.6 ou ultérieure.

Un schéma externe Amazon Redshift peut référencer une base de données dans un RDS MySQL ou Aurora MySQL. Dans ce cas, les limites suivantes s’appliquent :

- Lors de la création d'un schéma externe référençant un cluster de base de données Aurora, la base de données Aurora MySQL doit être de version 5.6 ou ultérieure.
- Lors de la création d’un schéma externe référençant Amazon RDS, la base de données RDS MySQL doit être à la version 5.6 ou ultérieure.

Amazon Redshift Spectrum

Cette section explique comment utiliser Redshift Spectrum pour lire efficacement les données d'Amazon S3.

Grâce à Amazon Redshift Spectrum, vous pouvez interroger et récupérer efficacement des données structurées et semi-structurées à partir de fichiers dans Amazon S3 sans avoir à charger les données dans des tables Amazon Redshift. Les requêtes Redshift Spectrum s’exécutent très rapidement sur de vastes jeux de données en appliquant le parallélisme massif. Une grande partie du traitement s’effectue dans la couche Spectre de Redshift, et la plupart des données restent dans Amazon S3. Plusieurs clusters peuvent interroger simultanément le même ensemble de données dans Amazon S3 sans devoir faire des copies des données pour chaque cluster.

Rubriques

- Présentation d’Amazon Redshift Spectrum
- Mise en route avec Amazon Redshift Spectrum
- Politiques IAM pour Amazon Redshift Spectrum
- Redshift Spectrum et AWS Lake Formation
- Fichiers de données pour les requêtes dans Amazon Redshift Spectrum
- Schémas externes dans Amazon Redshift Spectrum
- Tables externes pour Redshift Spectrum
- Utilisation de tables Apache Iceberg avec Amazon Redshift
- Performances des requêtes Amazon Redshift Spectrum
- Options de traitement des données
- Exemple : exécution de sous-requêtes corrélées dans Redshift Spectrum
- Mesures dans Amazon Redshift Spectrum
- Résolution des problèmes liés aux requêtes dans Amazon Redshift Spectrum
- Didacticiel : Faire une requête de données imbriquées avec Amazon Redshift Spectrum

Présentation d’Amazon Redshift Spectrum

Cette rubrique décrit les détails relatifs à l'utilisation de Redshift Spectrum pour lire efficacement depuis Amazon S3.

Amazon Redshift Spectrum réside sur des serveurs Amazon Redshift dédiés qui sont indépendants de votre cluster. Amazon Redshift transmet à la couche Redshift Spectrum de nombreuses tâches nécessitant une importante capacité de calcul, telles que le regroupement et le filtrage des prédicats. Ainsi, les requêtes Redshift Spectrum consomment nettement moins de capacité de traitement du cluster que les autres requêtes. Redshift Spectrum permet en outre un dimensionnement intelligent. Selon les demandes de vos requêtes, Redshift Spectrum est à même d'utiliser des milliers d'instances, afin de tirer parti du traitement massivement parallèle.

Pour créer des tables Redshift Spectrum, vous devez définir la structure de vos fichiers et enregistrer ces derniers en tant que tables dans un catalogue de données externe, Le catalogue de données externe peut être AWS Glue le catalogue de données fourni avec Amazon Athena ou votre propre métastore Apache Hive. Vous pouvez créer et gérer des tables externes soit à partir d'Amazon Redshift à l'aide de commandes DDL (data definition language), soit à l'aide de tout autre outil qui se connecte au catalogue de données externes. Les modifications apportées au catalogue de données externe sont immédiatement disponibles pour n'importe lequel de vos clusters Amazon Redshift.

Vous avez aussi la possibilité de partitionner les tables externes en une ou plusieurs colonnes, ce qui dans certains cas permet d'optimiser les performances, L'amélioration se produit parce que l'optimiseur de requêtes Amazon Redshift élimine les partitions qui ne contiennent pas de données pour la requête.

Les vues matérialisées sur les tables Spectrum peuvent considérablement améliorer les coûts et les performances. Pour de plus amples informations, veuillez consulter Vues matérialisées sur des tables de lacs de données externes dans Amazon Redshift Spectrum.

Une fois que vos tables Redshift Spectrum ont été définies, vous pouvez interroger et joindre les tables comme vous le faites avec n'importe quelle autre table Amazon Redshift. Redshift Spectrum ne prend pas en charge les opérations de mise à jour des tables externes. Vous pouvez ajouter des tables Redshift Spectrum à plusieurs clusters Amazon Redshift et interroger les mêmes données sur Amazon S3 à partir de n'importe quel cluster de la même région. AWS Lorsque vous mettez à jour des fichiers de données Amazon S3, les données sont immédiatement disponibles pour être interrogées à partir de n'importe lequel de vos clusters Amazon Redshift.

Le catalogue de AWS Glue données auquel vous accédez peut être crypté pour renforcer la sécurité. Si le AWS Glue catalogue est crypté, vous avez besoin de la clé AWS Key Management Service (AWS KMS) AWS Glue pour accéder au AWS Glue catalogue. AWS Glue le chiffrement du catalogue n'est pas disponible dans toutes les AWS régions. Pour obtenir la liste des AWS régions prises en charge, consultez la section Chiffrement et accès sécurisé AWS Glue dans le guide du AWS Glue

développeur.Pour plus d'informations sur le chiffrement du catalogue de AWS Glue données, voir Chiffrer votre catalogue de AWS Glue données dans le guide du AWS Glue développeur.

> **Note**
>
> Vous ne pouvez pas afficher les détails des tables Redshift Spectrum en utilisant les mêmes ressources que celles que vous utilisez pour les tables Amazon Redshift standard, telles que PG_TABLE_DEF, STV_TBL_PERM, PG_CLASS ou information_schema. Si votre outil de Business Intelligence ou d'analyse ne reconnaît pas les tables externes Redshift Spectrum, configurez votre application de façon à interroger SVV_EXTERNAL_TABLES et SVV_EXTERNAL_COLUMNS.

Régions Amazon Redshift Spectrum

Redshift Spectrum est disponible Régions AWS là où Amazon Redshift est disponible, sauf indication contraire dans la documentation spécifique à la région. Pour connaître Région AWS la disponibilité dans les régions commerciales, consultez la section Points de terminaison de service pour l'API Redshift dans le. Référence générale d'Amazon Web Services

Limites d'Amazon Redshift Spectrum

Cette rubrique décrit les limites d'utilisation de Redshift Spectrum.

Tenez compte des considérations suivantes lorsque vous utilisez Redshift Spectrum :

- Le cluster Amazon Redshift et le compartiment Amazon S3 doivent se trouver dans la même AWS région.
- Redshift Spectrum ne prend pas en charge le routage VPC amélioré avec des clusters provisionnés. Pour accéder à vos données Amazon S3, vous pouvez avoir besoin d'effectuer des étapes de configuration supplémentaires. Pour plus d'informations, consultez Utilisation d'Amazon Redshift Spectrum avec le routage VPC amélioré dans le Guide de gestion Amazon Redshift.
- Redshift Spectrum prend en charge les alias de point d'accès Amazon S3. Pour plus d'informations, consultez Utilisation d'un alias de type compartiment pour votre point d'accès dans le Guide de l'utilisateur Amazon Simple Storage Service. Cependant, Redshift Spectrum ne prend pas en charge le VPC avec les alias de point d'accès Amazon S3. Pour plus d'informations, consultez Utilisation d'Amazon Redshift Spectrum avec le routage VPC amélioré dans le Guide de gestion Amazon Redshift.

- Vous ne pouvez pas exécuter d'opérations de mise à jour ou de suppression sur les tables externes. Pour créer une table externe dans le schéma spécifié, vous pouvez utiliser CREATE EXTERNAL TABLE. Pour de plus amples informations sur la commande CREATE EXTERNAL TABLES, consultez CREATE EXTERNAL TABLE. Pour insérer les résultats d'une requête SELECT dans des tables externes existantes des catalogues externes, vous pouvez utiliser INSERT (table externe). Pour plus d'informations sur INSERT (table externe), consultez INSERT (table externe).
- À moins que vous n'utilisiez un AWS Glue Data Catalog qui soit activé pour AWS Lake Formation, vous ne pouvez pas contrôler les autorisations des utilisateurs sur une table externe. Vous pouvez en revanche accorder et révoquer des autorisations pour le schéma externe. Pour plus d'informations sur l'utilisation de AWS Lake Formation, consultezRedshift Spectrum et AWS Lake Formation.
- Pour exécuter des requêtes Redshift Spectrum, l'utilisateur de la base de données doit avoir l'autorisation d'y créer des tables temporaires. L'exemple suivant accorde une autorisation temporaire concernant la base de données `spectrumdb` au groupe d'utilisateurs `spectrumusers`.

  ```
  grant temp on database spectrumdb to group spectrumusers;
  ```

  Pour de plus amples informations, veuillez consulter GRANT.
- Lorsque vous utilisez le catalogue de données Athena ou le catalogue de AWS Glue données comme magasin de métadonnées, consultez la section Quotas et limites du guide de gestion Amazon Redshift.
- Redshift Spectrum ne prend pas en charge Amazon EMR avec Kerberos.

Mise en route avec Amazon Redshift Spectrum

Dans ce tutoriel, vous apprenez à utiliser Amazon Redshift Spectrum pour interroger des données directement à partir de fichiers sur Amazon S3. Si vous disposez déjà d'un cluster et d'un client SQL, vous pouvez effectuer ce tutoriel avec un effort de configuration minimal.

> **Note**
>
> Les requêtes Redshift Spectrum engendrent des frais supplémentaires. Le coût inhérent à l'exécution des exemples de requêtes de ce tutoriel est minime. Pour plus d'informations sur la tarification, consultez Tarification Amazon Redshift Spectrum.

Prérequis

Pour utiliser Redshift Spectrum, vous avez besoin d'un cluster Amazon Redshift et d'un client SQL qui est connecté à votre cluster afin que vous puissiez exécuter des commandes SQL. Le cluster et les fichiers de données dans Amazon S3 doivent se trouver dans la même Région AWS.

Pour plus d'informations sur la création d'un cluster Amazon Redshift, consultez la section Commencer avec les entrepôts de données provisionnés Amazon Redshift dans le guide de démarrage Amazon Redshift. Pour plus d'informations sur les méthodes de connexion à un cluster, consultez la section Connexion aux entrepôts de données Amazon Redshift dans le guide de démarrage Amazon Redshift.

Dans certains des exemples qui suivent, les données d'exemple se trouvent dans la région USA Est (Virginie du Nord) (`us-east-1`) et vous avez donc besoin d'un cluster qui figure également dans `us-east-1`. Vous pouvez également utiliser Amazon S3 pour copier des objets de données depuis les compartiments et dossiers suivants vers votre compartiment Région AWS où se trouve votre cluster :

- `s3://redshift-downloads/tickit/spectrum/customers/*`
- `s3://redshift-downloads/tickit/spectrum/sales_partition/*`
- `s3://redshift-downloads/tickit/spectrum/sales/*`
- `s3://redshift-downloads/tickit/spectrum/salesevent/*`

Exécutez une commande Amazon S3 similaire à la suivante pour copier les données d'exemple situées dans la région USA Est (Virginie du Nord) vers votre Région AWS. Avant d'exécuter cette commande, créez votre compartiment et vos dossiers dans votre compartiment pour qu'ils correspondent à votre commande de copie Amazon S3. La sortie de la commande de copie Amazon S3 confirme que les fichiers sont copiés *`bucket-name`* dans le fichier de votre choix Région AWS.

```
aws s3 cp s3://redshift-downloads/tickit/spectrum/ s3://bucket-name/tickit/spectrum/ --
copy-props none --recursive
```

Commencer à utiliser Redshift Spectrum en utilisant AWS CloudFormation

Comme alternative aux étapes suivantes, vous pouvez accéder au DataLake AWS CloudFormation modèle Redshift Spectrum pour créer une pile avec un compartiment Amazon S3 que vous pouvez interroger. Pour de plus amples informations, veuillez consulter Lancez votre AWS CloudFormation stack, puis interrogez vos données dans Amazon S3.

Mise en route avec Amazon Redshift Spectrum étape par étape

Suivez ces étapes pour commencer à utiliser Amazon Redshift Spectrum :

- Étape 1. Créer un rôle IAM pour Amazon Redshift
- Étape 2 : Association du rôle IAM au cluster
- Étape 3 : Création d'un schéma externe et d'une table externe
- Étape 4 : Interrogation de vos données dans Amazon S3

Étape 1. Créer un rôle IAM pour Amazon Redshift

Votre cluster a besoin d'une autorisation pour accéder à votre catalogue de données externe dans AWS Glue Amazon Athena et à vos fichiers de données dans Amazon S3. Pour fournir cette autorisation, vous référencez un rôle AWS Identity and Access Management (IAM) qui est attaché à votre cluster. Pour plus d'informations sur l'utilisation des rôles avec Amazon Redshift, consultez Autoriser les opérations de COPY et UNLOAD à l'aide des rôles IAM.

> **Note**
>
> Dans certains cas, vous pouvez migrer votre catalogue de données Athena vers un catalogue de AWS Glue données. Vous pouvez le faire si votre cluster se trouve dans une AWS région prise en charge et si le catalogue de données Athena contient des tables externes Redshift Spectrum. AWS Glue Pour utiliser le catalogue de AWS Glue données avec Redshift Spectrum, vous devrez peut-être modifier vos politiques IAM. Pour plus d'informations, consultez Mise à niveau vers le catalogue de données AWS Glue dans le Guide de l'utilisateur Athena.

Lorsque vous créez un rôle pour Amazon Redshift, choisissez une des approches suivantes :

- Si vous utilisez Redshift Spectrum avec un catalogue de données Athena ou AWS Glue un catalogue de données, suivez les étapes décrites dans. Pour créer un rôle IAM pour Amazon Redshift
- Si vous utilisez Redshift Spectrum avec un AWS Glue Data Catalog qui est activé pour AWS Lake Formation, suivez les étapes décrites dans les procédures suivantes :
  - Pour créer un rôle IAM pour Amazon Redshift à l'aide d'un AWS Glue Data CatalogAWS Lake Formation
  - Pour accorder des droits SELECT sur la table à interroger dans la base de données Lake Formation

Pour créer un rôle IAM pour Amazon Redshift

1. Ouvrez la console IAM.
2. Dans le panneau de navigation, choisissez Roles (Rôles).
3. Sélectionnez Create role (Créer un rôle).
4. Choisissez Service AWS comme entité de confiance, puis choisissez Redshift comme cas d'utilisation.
5. Sous Cas d'utilisation pour les autres Services AWS, choisissez Redshift - Personnalisable, puis cliquez sur Suivant.
6. La page Add permissions policy (Ajouter une politique d'autorisations) s'affiche. Choisissez `AmazonS3ReadOnlyAccess` et`AWSGlueConsoleFullAccess`, si vous utilisez le catalogue AWS Glue de données. Ou choisissez `AmazonAthenaFullAccess` si vous utilisez le catalogue de données Athena. Choisissez Suivant.

   > **Note**
   >
   > La politique `AmazonS3ReadOnlyAccess` accorde à votre cluster un accès en lecture seule à tous les compartiments Amazon S3. Pour accorder l'accès uniquement à l' AWS exemple de compartiment de données, créez une nouvelle politique et ajoutez les autorisations suivantes.

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

```
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": "arn:aws:s3:::redshift-downloads/*"
        }
    ]
}
```

7. Pour Nom du rôle, indiquez le nom de votre rôle, par exemple **`myspectrum_role`**.
8. Passez en revue les informations, puis choisissez Créer un rôle.
9. Dans le panneau de navigation, choisissez Roles (Rôles). Choisissez le nom de votre nouveau rôle pour afficher le récapitulatif, puis copiez l'ARN de rôle dans le presse-papiers. Il s'agit de l'ARN (Amazon Resource Name) du rôle que vous venez de créer. Vous utilisez cette valeur lorsque vous créez des tables externes pour référencer vos fichiers de données sur Amazon S3.

Pour créer un rôle IAM pour Amazon Redshift à l'aide d'un AWS Glue Data CatalogAWS Lake Formation

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation, choisissez Politiques.

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s'affiche. Sélectionnez Get started (Mise en route).
3. Choisissez Create Policy (Créer une politique).
4. Choisissez de créer la politique dans l'onglet JSON.
5. Collez le document de politique JSON suivant, qui accorde l'accès au catalogue de données mais refuse les autorisations d'administrateur pour Lake Formation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RedshiftPolicyForLF",
            "Effect": "Allow",
            "Action": [
                "glue:*",
                "lakeformation:GetDataAccess"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

6. Lorsque vous avez terminé, choisissez Review (Vérifier) pour vérifier la politique. Le programme de validation des politiques signale les éventuelles erreurs de syntaxe.
7. Sur la page Vérifier la politique, dans le champ Nom, saisissez **`myspectrum_policy`** pour nommer la politique que vous créez. (Facultatif) Entrez une description. Vérifiez le récapitulatif de politique pour voir les autorisations accordées par votre politique. Sélectionnez ensuite Créer une politique pour enregistrer votre travail.

   Une fois que vous avez créé une politique, vous pouvez fournir un accès à vos utilisateurs.

Pour activer l'accès, ajoutez des autorisations à vos utilisateurs, groupes ou rôles :

- Utilisateurs et groupes dans AWS IAM Identity Center :

  Créez un jeu d'autorisations. Suivez les instructions de la rubrique Création d'un jeu d'autorisations du Guide de l'utilisateur AWS IAM Identity Center .
- Utilisateurs gérés dans IAM par un fournisseur d'identité :

  Créez un rôle pour la fédération d'identité. Suivez les instructions de la rubrique Création d'un rôle pour un fournisseur d'identité tiers (fédération) dans le Guide de l'utilisateur IAM.
- Utilisateurs IAM :
  - Créez un rôle que votre utilisateur peut assumer. Suivez les instructions de la rubrique Création d'un rôle pour un utilisateur IAM dans le Guide de l'utilisateur IAM.
  - (Non recommandé) Attachez une politique directement à un utilisateur ou ajoutez un utilisateur à un groupe d'utilisateurs. Suivez les instructions de la rubrique Ajout d'autorisations à un utilisateur (console) du Guide de l'utilisateur IAM.

Pour accorder des droits SELECT sur la table à interroger dans la base de données Lake Formation

1. Ouvrez la console Lake Formation à l'adresse https://console.aws.amazon.com/lakeformation/.
2. Dans le volet de navigation, sélectionnez Autorisations de lac de données, puis Accorder.

3. Suivez les instructions de la page Octroi d'autorisations de table à l'aide de la méthode de ressource nommée dans le Guide du développeur AWS Lake Formation . Saisissez les informations suivantes :

   - Pour le rôle IAM, choisissez celui que vous avez créé, `myspectrum_role`. Lorsque vous exécutez l'éditeur de requêtes Amazon Redshift, il utilise ce rôle IAM pour l'autorisation des données.

   > **Note**
   >
   > Pour accorder l'autorisation SELECT sur la table d'un catalogue de données Lake Formation afin d'interroger, procédez comme suit :
   >
   > - Enregistrez le chemin d'accès aux données dans Lake Formation.
   > - Accordez les autorisations utilisateur sur ce chemin dans Lake Formation..
   > - Les tables créées se trouvent dans le chemin enregistré dans Lake Formation..

4. Choisissez Accorder.

> **Important**
>
> La bonne pratique consiste à n'autoriser l'accès qu'aux objets Amazon S3 sous-jacents par le biais des autorisations Lake Formation. Pour empêcher tout accès non approuvé, supprimez toute autorisation accordée aux objets Amazon S3 en dehors de Lake Formation. Si vous accédiez précédemment aux objets Amazon S3 avant de configurer Lake Formation, supprimez toutes les politiques IAM ou les autorisations de compartiment qui étaient précédemment configurées. Pour plus d'informations, consultez les sections Mise à niveau AWS Glue des autorisations de données vers les autorisations du AWS Lake Formation modèle et de Lake Formation.

Étape 2 : Association du rôle IAM au cluster

Vous avez maintenant un rôle IAM qui autorise Amazon Redshift à accéder au catalogue de données externe et à Amazon S3 pour vous. At this point, you must associate that role with your Amazon Redshift cluster.

Pour associer un rôle IAM à un cluster

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis le nom du cluster que vous souhaitez mettre à jour.
3. Sous Actions, choisissez Gérer les rôles IAM. La page Rôles IAM apparaît.
4. Choisissez Enter ARN (Entrer l'ARN), puis entrez un ARN ou un rôle IAM, ou choisissez un rôle IAM dans la liste. Choisissez ensuite Ajouter un rôle IAM pour l'ajouter à la liste des Rôles IAM attachés.
5. Choisissez Terminé pour associer le rôle IAM au cluster. Le cluster est modifié pour finaliser la modification.

Étape 3 : Création d'un schéma externe et d'une table externe

Créez des tables externes dans un schéma externe. Le schéma externe référence une base de données dans le catalogue de données externe et fournit le rôle IAM ARN qui autorise votre cluster à accéder à Amazon S3 en votre nom. Vous pouvez créer une base de données externe dans un catalogue de données Amazon Athena ou dans un métastore Apache Hive, tel qu'Amazon EMR. AWS Glue Data Catalog Pour cet exemple, vous créez la base de données externe dans un catalogue de données Amazon Athena lorsque vous créez le schéma externe Amazon Redshift. Pour de plus amples informations, consultez Schémas externes dans Amazon Redshift Spectrum.

Pour créer un schéma externe et une table externe

1. Pour créer un schéma externe, remplacez l'ARN de rôle IAM dans la commande ci-après par l'ARN de rôle que vous avez créé à l'étape 1. Ensuite, exécutez la commande dans votre client SQL.

```
create external schema myspectrum_schema
from data catalog
database 'myspectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myspectrum_role'
create external database if not exists;
```

2. Pour créer une table externe, exécutez la commande CREATE EXTERNAL TABLE suivante.

Note

Votre cluster et le compartiment Amazon S3 doivent se trouver dans la même Région AWS. Pour cet exemple de commande CREATE EXTERNAL TABLE, le compartiment Amazon S3 contenant les exemples de données est situé dans l'est des États-Unis (Virginie du Nord) Région AWS. Pour voir les données sources, téléchargez le fichier `sales_ts.000`. .

Vous pouvez modifier cet exemple pour l'exécuter dans un autre Région AWS. Créez un compartiment Amazon S3 dans le compartiment de votre choix Région AWS. Copiez les données de vente à l'aide d'une commande de copie Amazon S3. Ensuite, mettez à jour l'option d'emplacement dans l'exemple de commande `CREATE EXTERNAL TABLE` sur votre compartiment.

```
aws s3 cp s3://redshift-downloads/tickit/spectrum/sales/ s3://bucket-name/
tickit/spectrum/sales/ --copy-props none --recursive
```

Le résultat de la commande de copie Amazon S3 confirme que le fichier a été copié *bucket-name* dans le fichier de votre choix Région AWS.

```
copy: s3://redshift-downloads/tickit/spectrum/sales/sales_ts.000 to
 s3://bucket-name/tickit/spectrum/sales/sales_ts.000
```

```
create external table myspectrum_schema.sales(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales/'
```

```
table properties ('numRows'='172000');
```

Étape 4 : Interrogation de vos données dans Amazon S3

Une fois vos tables externes créées, vous pouvez les interroger à l'aide des mêmes instructions SELECT que vous utilisez pour interroger d'autres tables Amazon Redshift. Ces requêtes d'instruction SELECT incluent la jonction des tables, le regroupement des données et le filtrage des prédicats.

Pour interroger vos données dans Amazon S3

1. Obtenez le nombre de lignes dans la table MYSPECTRUM_SCHEMA.SALES.

```
select count(*) from myspectrum_schema.sales;
```

```
count
------
172462
```

2. En guise de bonne pratique, gardez vos plus grandes tables de faits dans Amazon S3 et vos plus petites tables de dimensions dans Amazon Redshift. Si vous avez chargé les exemples de données dans Charger des données, vous disposez d'une table nommée EVENT dans votre base de données. Sinon, créez la table EVENT en exécutant la commande suivante.

```
create table event(
eventid integer not null distkey,
venueid smallint not null,
catid smallint not null,
dateid smallint not null sortkey,
eventname varchar(200),
starttime timestamp);
```

3. Chargez la table EVENT en remplaçant l'ARN de rôle IAM dans la commande COPY ci-après par l'ARN de rôle que vous avez créé dans Étape 1. Créer un rôle IAM pour Amazon Redshift. Vous pouvez éventuellement télécharger et consulter les données source pour le `allevents_pipe.txt` depuis un compartiment Amazon S3 dans Région AWS `us-east-1`.

```
copy event from 's3://redshift-downloads/tickit/allevents_pipe.txt'
iam_role 'arn:aws:iam::123456789012:role/myspectrum_role'
```

```
delimiter '|' timeformat 'YYYY-MM-DD HH:MI:SS' region 'us-east-1';
```

L'exemple suivant joint la table Amazon S3 externe MYSPECTRUM_SCHEMA.SALES à la table Amazon Redshift locale EVENT afin de déterminer le total des ventes pour les 10 principaux événements.

```
select top 10 myspectrum_schema.sales.eventid,
 sum(myspectrum_schema.sales.pricepaid) from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
    289 | 51846.00
   7895 | 51049.00
   1602 | 50301.00
    851 | 49956.00
   7315 | 49823.00
   6471 | 47997.00
   2118 | 47863.00
    984 | 46780.00
   7851 | 46661.00
   5638 | 46280.00
```

4. Affichez le plan de la requête précédente. Observez les étapes `S3 Seq Scan`, `S3 HashAggregate` et `S3 Query Scan` qui ont été exécutées par rapport aux données sur Amazon S3.

```
explain
select top 10 myspectrum_schema.sales.eventid,
 sum(myspectrum_schema.sales.pricepaid)
from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
QUERY PLAN


------------------------------------------------------------------------------
XN Limit  (cost=1001055770628.63..1001055770628.65 rows=10 width=31)


  ->  XN Merge  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


        Merge Key: sum(sales.derived_col2)


        ->  XN Network  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)


              Send to leader


              ->  XN Sort  (cost=1001055770628.63..1001055770629.13 rows=200
 width=31)

                    Sort Key: sum(sales.derived_col2)


                    ->  XN HashAggregate  (cost=1055770620.49..1055770620.99
 rows=200 width=31)

                          ->  XN Hash Join DS_BCAST_INNER
 (cost=3119.97..1055769620.49 rows=200000 width=31)

                                Hash Cond: ("outer".derived_col1 = "inner".eventid)


                                ->  XN S3 Query Scan sales  (cost=3010.00..5010.50
 rows=200000 width=31)

                                      ->  S3 HashAggregate  (cost=3010.00..3010.50
 rows=200000 width=16)

                                            ->  S3 Seq Scan myspectrum_schema.sales
 location:"s3://redshift-downloads/tickit/spectrum/sales" format:TEXT
 (cost=0.00..2150.00 rows=172000 width=16)
```

```
                                                Filter: (pricepaid > 30.00)

                                ->  XN Hash  (cost=87.98..87.98 rows=8798 width=4)

                                      ->  XN Seq Scan on event  (cost=0.00..87.98
rows=8798 width=4)
```

Lancez votre AWS CloudFormation stack, puis interrogez vos données dans Amazon S3

Une fois que vous avez créé un cluster Amazon Redshift et que vous vous y êtes connecté, vous pouvez installer votre DataLake AWS CloudFormation modèle Redshift Spectrum, puis interroger vos données.

CloudFormation installe le modèle Redshift Spectrum Getting DataLake Started et crée une pile contenant les éléments suivants :

- Un rôle nommé `myspectrum_role` associé à votre cluster Redshift
- Un schéma externe nommé `myspectrum_schema`
- Une table externe nommée `sales` dans un compartiment Amazon S3
- Une table Redshift nommée `event` chargée de données

Pour lancer votre stack Redshift Spectrum Getting Started DataLake CloudFormation

1. Choisissez Launch CFN stack (Lancer la pile CFN). La CloudFormation console s'ouvre avec le modèle DataLake .yml sélectionné.

   Vous pouvez également télécharger et personnaliser le modèle DataLake CloudFormation CFN Redshift Spectrum Getting Started, puis ouvrir la CloudFormation console (https://console.aws.amazon.com/cloudformation) et créer une pile avec le modèle personnalisé.
2. Choisissez Suivant.
3. Sous Parameters (Paramètres), saisissez le nom du cluster Amazon Redshift, le nom de base de données et le nom d'utilisateur de votre base de données.
4. Choisissez Suivant.

Les options de pile apparaissent.

5. Choisissez Next (Suivant) pour accepter les paramètres par défaut.
6. Passez en revue les informations et sous Fonctionnalités, puis sélectionnez Je reconnais que cela AWS CloudFormation pourrait créer des ressources IAM.
7. Sélectionnez Créer la pile.

Si une erreur se produit pendant la création de la pile, consultez les informations suivantes :

- Consultez l'onglet CloudFormation Événements pour obtenir des informations qui peuvent vous aider à résoudre l'erreur.
- Supprimez la DataLake CloudFormation pile avant de recommencer l'opération.
- Assurez-vous que vous êtes connecté à votre base de données Amazon Redshift.
- Assurez-vous d’avoir saisi les informations correctes pour le nom de cluster Amazon Redshift, le nom de base de données et le nom d’utilisateur de la base de données.

Interrogez vos données dans Amazon S3

Vous interrogez les tables externes à l’aide des mêmes instructions SELECT que vous utilisez pour interroger d’autres tables Amazon Redshift. Ces requêtes d’instruction SELECT incluent la jonction des tables, le regroupement des données et le filtrage des prédicats.

La requête suivante renvoie le nombre de lignes dans la table externe `myspectrum_schema.sales`.

```
select count(*) from myspectrum_schema.sales;
```

```
count
------
172462
```

Joindre une table externe à une table locale

L’exemple suivant joint la table externe `myspectrum_schema.sales` à la table locale `event` afin de déterminer le total des ventes pour les 10 principaux événements.

```
select top 10 myspectrum_schema.sales.eventid, sum(myspectrum_schema.sales.pricepaid)
 from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
    289 | 51846.00
   7895 | 51049.00
   1602 | 50301.00
    851 | 49956.00
   7315 | 49823.00
   6471 | 47997.00
   2118 | 47863.00
    984 | 46780.00
   7851 | 46661.00
   5638 | 46280.00
```

Afficher le plan de requêtes

Affichez le plan de la requête précédente. Observez les étapes `S3 Seq Scan`, `S3 HashAggregate` et `S3 Query Scan` qui ont été exécutées par rapport aux données sur Amazon S3.

```
explain
select top 10 myspectrum_schema.sales.eventid, sum(myspectrum_schema.sales.pricepaid)
from myspectrum_schema.sales, event
where myspectrum_schema.sales.eventid = event.eventid
and myspectrum_schema.sales.pricepaid > 30
group by myspectrum_schema.sales.eventid
order by 2 desc;
```

```
QUERY PLAN


------------------------------------------------------------------------------
XN Limit  (cost=1001055770628.63..1001055770628.65 rows=10 width=31)
```

```
 ->  XN Merge  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)

       Merge Key: sum(sales.derived_col2)

       ->  XN Network  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)

             Send to leader

             ->  XN Sort  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)

                   Sort Key: sum(sales.derived_col2)

                   ->  XN HashAggregate  (cost=1055770620.49..1055770620.99 rows=200
width=31)

                         ->  XN Hash Join DS_BCAST_INNER  (cost=3119.97..1055769620.49
rows=200000 width=31)

                               Hash Cond: ("outer".derived_col1 = "inner".eventid)

                               ->  XN S3 Query Scan sales  (cost=3010.00..5010.50
rows=200000 width=31)

                                     ->  S3 HashAggregate  (cost=3010.00..3010.50
rows=200000 width=16)

                                           ->  S3 Seq Scan spectrum.sales
location:"s3://redshift-downloads/tickit/spectrum/sales" format:TEXT
(cost=0.00..2150.00 rows=172000 width=16)
                                                 Filter: (pricepaid > 30.00)

                               ->  XN Hash  (cost=87.98..87.98 rows=8798 width=4)
```

```
                                  ->  XN Seq Scan on event  (cost=0.00..87.98
rows=8798 width=4)
```

Politiques IAM pour Amazon Redshift Spectrum

Cette rubrique décrit les autorisations IAM requises pour utiliser Redshift Spectrum.

Par défaut, Amazon Redshift Spectrum utilise AWS Glue Data Catalog les régions prises AWS en charge AWS Glue. Dans d'autres AWS régions, Redshift Spectrum utilise le catalogue de données Athena. Votre cluster a besoin d'une autorisation pour accéder à votre catalogue de données externe dans AWS Glue ou Athena et à vos fichiers de données dans Amazon S3. Vous fournissez cette autorisation en faisant référence à un rôle AWS Identity and Access Management (IAM) attaché à votre cluster. Si vous utilisez un métastore Apache Hive pour gérer votre catalogue de données, il n'est pas nécessaire que vous fournissiez un accès à Athena.

Vous pouvez créer des chaînes de rôles pour permettre à votre cluster d'endosser d'autres rôles non attachés au cluster. Pour de plus amples informations, veuillez consulter Créer des rôles IAM dans Amazon Redshift Spectrum.

Le AWS Glue catalogue auquel vous accédez peut être crypté pour renforcer la sécurité. Si le AWS Glue catalogue est crypté, vous avez besoin de la AWS KMS clé AWS Glue pour accéder au catalogue de AWS Glue données. Pour plus d'informations, consultez la section Chiffrer votre catalogue de AWS Glue données dans le guide du AWS Glue développeur.

Rubriques

- Autorisations Amazon S3
- Autorisations Amazon S3 entre comptes
- Politiques visant à accorder ou restreindre l'accès à l'aide de Redshift Spectrum
- Politiques visant à accorder des autorisations minimales
- Créer des rôles IAM dans Amazon Redshift Spectrum
- Contrôle de l'accès au catalogue AWS Glue de données

Autorisations Amazon S3

Au minimum, votre cluster a besoin d'un accès GET et LIST à votre compartiment Amazon S3. Si votre bucket n'est pas dans le même AWS compte que votre cluster, celui-ci doit également autoriser

votre cluster à accéder aux données. Pour plus d'informations, consultez Autoriser Amazon Redshift à accéder à AWS d'autres services en votre nom.

> **Note**
>
> Le compartiment Amazon S3 ne peut pas appliquer une politique de compartiment qui restreint l'accès uniquement à partir de points de terminaison VPC spécifiques.

La politique suivante accorde un accès GET et LIST à n'importe quel compartiment Amazon S3. La politique autorise l'accès aux compartiments Amazon S3 pour Redshift Spectrum, ainsi que les opérations COPY.

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": ["s3:Get*", "s3:List*"],
  "Resource": "*"
 }]
}
```

La politique suivante accorde un accès GET et LIST à votre compartiment Amazon S3 nommé `amzn-s3-demo-bucket`.

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

Autorisations Amazon S3 entre comptes

Pour autoriser Redshift Spectrum à accéder aux données d'un compartiment Amazon S3 appartenant à un autre AWS compte, ajoutez la politique suivante au compartiment Amazon S3. Pour plus d'informations, consultez Accorder des autorisations entre comptes sur un compartiment.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Example permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::redshift-account:role/spectrumrole"
            },
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListMultipartUploadParts",
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": [
                "arn:aws:s3:::bucketname",
                "arn:aws:s3:::bucketname/*"
            ]
        }
    ]
}
```

Politiques visant à accorder ou restreindre l'accès à l'aide de Redshift Spectrum

L'exemple de politique suivant autorise l'accès aux compartiments Amazon S3 pour Amazon Redshift. Lorsque vous ajoutez cette politique à un rôle IAM, nous vous recommandons d'utiliser un rôle IAM uniquement utilisé pour Amazon Redshift. Nous vous recommandons également de ne pas réutiliser les rôles existants ni d'ajouter d'autorisations supplémentaires à ce rôle IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucketVersions",
                "s3:ListBucket"
```

```
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

Politiques visant à accorder des autorisations minimales

La politique suivante accorde les autorisations minimales requises pour utiliser Redshift Spectrum avec Amazon S3 et AWS Glue Athena.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucketname",
                "arn:aws:s3:::bucketname/folder1/folder2/*"
             ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateDatabase",
                "glue:DeleteDatabase",
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:UpdateDatabase",
                "glue:CreateTable",
                "glue:DeleteTable",
                "glue:BatchDeleteTable",
                "glue:UpdateTable",
                "glue:GetTable",
                "glue:GetTables",
```

```
                "glue:BatchCreatePartition",
                "glue:CreatePartition",
                "glue:DeletePartition",
                "glue:BatchDeletePartition",
                "glue:UpdatePartition",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition"
            ],
            "Resource": "*"
        }
    ]
}
```

Si vous utilisez Athena pour votre catalogue de données à la place de AWS Glue, la politique exige un accès complet à Athéna. La politique suivante donne accès aux ressources Athena. Si votre base de données externe se trouve dans un métastore Hive, vous n'avez pas besoin de l'accès Athena.

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": ["athena:*"],
  "Resource": ["*"]
 }]
}
```

Créer des rôles IAM dans Amazon Redshift Spectrum

Lorsque vous attachez un rôle à votre cluster, celui-ci peut assumer ce rôle pour accéder à Amazon S3, Athena et en votre AWS Glue nom. Si un rôle attaché à votre cluster n'a pas accès aux ressources nécessaires, vous pouvez créer une chaîne avec un autre rôle, lequel peut appartenir à un autre compte. Votre cluster endosse alors provisoirement le rôle relié par la chaîne afin d'accéder aux données. Vous pouvez également accorder des accès entre comptes en créant des chaînes de rôles. Une chaîne comprend 10 rôles maximum. Chaque rôle de la chaîne passe au rôle suivant, jusqu'à ce que le cluster endosse le dernier rôle de la chaîne.

Pour créer une chaîne de rôles, vous devez établir une relation d'approbation entre ces rôles. Un rôle endossant un autre rôle doit avoir une politique d'autorisation lui permettant d'endosser le rôle spécifié. Le rôle qui transmet les autorisations, quant à lui, doit avoir une politique d'approbation

lui permettant de transmettre ses autorisations à un autre rôle. Pour plus d'informations, consultez Création de liens de rôles IAM dans Amazon Redshift.

Lorsque vous exécutez la commande CREATE EXTERNAL SCHEMA, vous pouvez enchaîner les rôles en incluant une liste de rôles séparés par des virgules. ARNs

> **Note**
>
> La liste des rôles de la chaîne ne doit pas inclure d'espaces.

Dans l'exemple suivant, `MyRedshiftRole` est attaché au cluster. `MyRedshiftRole` assume le rôle `AcmeData` qui appartient au compte `111122223333`.

```
create external schema acme from data catalog
database 'acmedb' region 'us-west-2'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole,arn:aws:iam::111122223333:role/
AcmeData';
```

Contrôle de l'accès au catalogue AWS Glue de données

Si vous l'utilisez AWS Glue pour votre catalogue de données, vous pouvez appliquer un contrôle d'accès précis au catalogue de AWS Glue données dans le cadre de votre politique IAM. Par exemple, vous pouvez exposer uniquement quelques bases de données et tables à un rôle IAM spécifique.

Les sections suivantes décrivent les politiques IAM pour les différents niveaux d'accès aux données stockées dans le catalogue de AWS Glue données.

Rubriques

- Politique pour les opérations de base de données
- Politique pour les opérations de table
- Politique pour des opérations de partition

Politique pour les opérations de base de données

Si vous souhaitez autoriser les utilisateurs à consulter et à créer une base de données, ils doivent disposer de droits d'accès à la fois à la base de données et au catalogue de AWS Glue données.

L'exemple de requête suivant crée une base de données.

```
CREATE EXTERNAL SCHEMA example_db
FROM DATA CATALOG DATABASE 'example_db' region 'us-west-2'
IAM_ROLE 'arn:aws:iam::redshift-account:role/spectrumrole'
CREATE EXTERNAL DATABASE IF NOT EXISTS
```

La politique IAM suivante fournit les autorisations minimales requises pour créer une base de données.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:CreateDatabase"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:catalog"
            ]
        }
    ]
}
```

L'exemple de requête suivant dresse la liste des bases de données actuelles.

```
SELECT * FROM SVV_EXTERNAL_DATABASES WHERE
databasename = 'example_db1' or databasename = 'example_db2';
```

La politique IAM suivante fournit les autorisations minimales requises pour répertorier les bases de données actuelles.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabases"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:database/example_db1",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db2",
                "arn:aws:glue:us-west-2:redshift-account:catalog"

            ]
        }
    ]
}
```

Politique pour les opérations de table

Si vous souhaitez octroyer aux utilisateurs les autorisations nécessaires pour créer, supprimer, modifier ou effectuer toute autre action sur les tables, ils ont besoin de plusieurs types d'accès. Ils ont besoin d'accéder aux tables elles-mêmes, aux bases de données auxquelles elles appartiennent et au catalogue.

L'exemple de requête suivant crée une table externe.

```
CREATE EXTERNAL TABLE example_db.example_tbl0(
    col0 INT,
    col1 VARCHAR(255)
) PARTITIONED BY (part INT) STORED AS TEXTFILE
LOCATION 's3://test/s3/location/';
```

La politique IAM suivante fournit les autorisations minimales requises pour créer une table externe.

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

```
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateTable"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

Les exemples de requêtes suivants dressent la liste des tables externes actuelles.

```
SELECT * FROM svv_external_tables
WHERE tablename = 'example_tbl0' OR
tablename = 'example_tbl1';
```

```
SELECT * FROM svv_external_columns
WHERE tablename = 'example_tbl0' OR
tablename = 'example_tbl1';
```

```
SELECT parameters FROM svv_external_tables
WHERE tablename = 'example_tbl0' OR
tablename = 'example_tbl1';
```

La politique IAM suivante fournit les autorisations minimales requises pour répertorier les tables externes actuelles.

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/
example_tbl0",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl1"

            ]
        }
    ]
}
```

L’exemple de requête suivant modifie une table existante.

```
ALTER TABLE example_db.example_tbl0
SET TABLE PROPERTIES ('numRows' = '100');
```

La politique IAM suivante fournit les autorisations minimales requises pour modifier une table existante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTable",
                "glue:UpdateTable"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
```

```
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"

            ]
        }
    ]
}
```

L'exemple de requête suivant supprime une table existante.

```
DROP TABLE example_db.example_tbl0;
```

La politique IAM suivante fournit les autorisations minimales requises pour supprimer une table existante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:DeleteTable"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

Politique pour des opérations de partition

Si vous souhaitez accorder à des utilisateurs des autorisations pour effectuer des opérations au niveau des partitions (afficher, créer, supprimer, modifier, etc.), ceux-ci ont besoin d'autorisations sur les tables auxquelles les partitions appartiennent. Ils ont également besoin d'autorisations sur les bases de données associées et le catalogue de données AWS Glue .

L'exemple de requête suivant crée une partition.

```
ALTER TABLE example_db.example_tbl0
ADD PARTITION (part=0) LOCATION 's3://test/s3/location/part=0/';
ALTER TABLE example_db.example_t
ADD PARTITION (part=1) LOCATION 's3://test/s3/location/part=1/';
```

La politique IAM suivante fournit les autorisations minimales requises pour créer une partition.

```
 {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetTable",
                "glue:BatchCreatePartition"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

L'exemple de requête suivant dresse la liste des partitions actuelles.

```
SELECT * FROM svv_external_partitions
WHERE schemname = 'example_db' AND
tablename = 'example_tbl0'
```

La politique IAM suivante fournit les autorisations minimales requises pour répertorier les partitions actuelles.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetPartitions",
                "glue:GetTables",
                "glue:GetTable"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

L'exemple de requête suivant modifie une partition existante.

```
ALTER TABLE example_db.example_tbl0 PARTITION(part='0')
SET LOCATION 's3://test/s3/new/location/part=0/';
```

La politique IAM suivante fournit les autorisations minimales requises pour modifier une partition existante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetPartition",
                "glue:UpdatePartition"
```

```
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

L'exemple de requête suivant supprime une partition existante.

```
ALTER TABLE example_db.example_tbl0 DROP PARTITION(part='0');
```

La politique IAM suivante fournit les autorisations minimales requises pour supprimer une partition existante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:DeletePartition"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:redshift-account:catalog",
                "arn:aws:glue:us-west-2:redshift-account:database/example_db",
                "arn:aws:glue:us-west-2:redshift-account:table/example_db/example_tbl0"
            ]
        }
    ]
}
```

Redshift Spectrum et AWS Lake Formation

Cette rubrique explique comment utiliser Redshift Spectrum avec Lake Formation. Lake Formation est un service de partage de données analytiques.

Vous pouvez l'utiliser AWS Lake Formation pour définir et appliquer de manière centralisée des politiques d'accès au niveau des bases de données, des tables et des colonnes aux données stockées dans Amazon S3. Une fois que vos données sont enregistrées sur un AWS Glue Data Catalog activé avec Lake Formation, vous pouvez les interroger en utilisant plusieurs services, dont Redshift Spectrum.

Lake Formation fournit la sécurité et la gouvernance du catalogue de données. Dans Lake Formation, vous pouvez accorder et révoquer des autorisations pour les objets du catalogue de données, tels que les bases de données, les tables, les colonnes et le stockage Amazon S3 sous-jacent.

> **Important**
>
> Vous ne pouvez utiliser Redshift Spectrum avec un catalogue de données compatible avec Lake Formation que dans AWS les régions où Lake Formation est disponible. Pour obtenir la liste des régions disponibles, consultez Points de terminaison et quotas AWS Lake Formation dans le Références générales AWS.

En utilisant Redshift Spectrum avec Lake Formation, vous pouvez effectuer les opérations suivantes :

- Utilisez Lake Formation en tant qu'emplacement centralisé dans lequel vous accordez et révoquez des autorisations et des politiques de contrôle d'accès sur toutes vos données du lac de données. Lake Formation fournit une hiérarchie d'autorisations pour contrôler l'accès aux bases de données et aux tables dans un catalogue de données. Pour plus d'informations, consultez Octroi d'autorisations Lake Formation du Guide du développeur AWS Lake Formation .
- Créer des tables externes et interroger les données du lac de données. Avant que les utilisateurs de votre compte puissent exécuter des requêtes, un administrateur de compte de lac de données enregistre vos chemins Amazon S3 existants contenant des données source avec Lake Formation. L'administrateur crée des tables et accorde des autorisations à vos utilisateurs. Des accès peuvent être donnés sur des bases de données, des tables ou des colonnes. L'administrateur peut utiliser des filtres de données dans Lake Formation pour accorder un contrôle d'accès précis à vos données sensibles stockées dans Amazon S3. Pour de plus amples informations, veuillez consulter Utilisation de filtres de données pour la sécurité au niveau de la ligne et au niveau de la cellule.

Une fois que les données sont enregistrées dans le catalogue de données, à chaque fois que les utilisateurs tentent d'exécuter des interrogations, Lake Formation vérifie l'accès à la table pour ce principal spécifique. Lake Formation attribue des informations d'identification temporaires à Redshift Spectrum et la requête s'exécute.

- Exécutez des requêtes Redshift Spectrum sur un appareil monté automatiquement à AWS Glue Data Catalog l'aide des informations d'identification IAM obtenues avec `GetCredentials` ou, et `GetClusterCredentials` gérez les autorisations de Lake Formation par utilisateur de base de données (IAMR:UserName ou IAM:UserName).

Lorsque vous utilisez Redshift Spectrum avec un catalogue de données activé pour Lake Formation, l'un des éléments suivants doit être en place :

- Rôle IAM associé au cluster autorisé à accéder au catalogue de données.
- Identité IAM fédérée configurée pour gérer l'accès aux ressources externes. Pour plus d'informations, consultez Utilisation d'une identité fédérée pour gérer l'accès d'Amazon Redshift aux ressources locales et aux tables externes Amazon Redshift Spectrum.

> **Important**
>
> Vous ne pouvez pas lier les rôles IAM lorsque vous utilisez Redshift Spectrum avec un catalogue de données Lake Formation.

Pour en savoir plus sur les étapes nécessaires à la configuration AWS Lake Formation en vue de l'utilisation de Redshift Spectrum, voir Tutoriel : Création d'un lac de données à partir d'une source JDBC dans Lake Formation du guide du développeur.AWS Lake Formation Plus précisément, consultez Interroger les données du lac de données à l'aide d'Amazon Redshift Spectrum pour obtenir des détails sur l'intégration avec Redshift Spectrum. Les données et AWS les ressources utilisées dans cette rubrique dépendent des étapes précédentes du didacticiel.

Utilisation de filtres de données pour la sécurité au niveau de la ligne et au niveau de la cellule

Vous pouvez définir des filtres de données AWS Lake Formation pour contrôler l'accès au niveau des lignes et des cellules de vos requêtes Redshift Spectrum aux données définies dans votre catalogue de données. Pour les configurer, vous effectuez les tâches suivantes :

- Créez un filtre de données dans Lake Formation avec les informations suivantes :
  - Spécification de colonne avec une liste de colonnes à inclure ou à exclure des résultats de la requête.
  - Expression de filtre de ligne qui spécifie les lignes à inclure dans les résultats de la requête.

  Pour plus d'informations sur la création d'un filtre de données, consultez Filtres de données dans Lake Formation dans le Manuel du développeur AWS Lake Formation .
- Créez une table externe dans Amazon Redshift qui fait référence à une table de votre catalogue de données compatible avec Lake Formation. Pour plus de détails sur la façon d'interroger une table de Lake Formation à l'aide de Redshift Spectrum, consultez Interroger les données du lac de données à l'aide d'Amazon Redshift Spectrum dans le Manuel du développeur AWS Lake Formation .

Une fois la table définie dans Amazon Redshift, vous pouvez interroger la table Lake Formation et accéder uniquement aux lignes et aux colonnes autorisées par le filtre de données.

Pour un guide détaillé sur la façon de configurer la sécurité au niveau des lignes et des cellules dans Lake Formation, puis d'effectuer des requêtes à l'aide de Redshift Spectrum, consultez Utiliser Amazon Redshift Spectrum avec des stratégies de sécurité au niveau des lignes et des cellules définies dans AWS Lake Formation.

Fichiers de données pour les requêtes dans Amazon Redshift Spectrum

Cette section décrit comment créer des fichiers de données dans Amazon S3 dans un format pris en charge par Redshift Spectrum.

Les fichiers de données que vous utilisez pour les requêtes dans Amazon Redshift Spectrum sont généralement les mêmes types de fichiers que vous utilisez pour d'autres applications. Par exemple,

les mêmes types de fichiers sont utilisés avec Amazon Athena, Amazon EMR et. QuickSight Vous pouvez interroger les données dans leur format original directement depuis Amazon S3. Pour ce faire, les fichiers de données doivent être dans un format pris en charge par Redshift Spectrum et être situés dans un compartiment Amazon S3 auquel votre cluster peut accéder.

Le compartiment Amazon S3 contenant les fichiers de données et le cluster Amazon Redshift doivent se trouver dans la même AWS région. Pour plus d'informations sur AWS les régions prises en charge, consultezRégions Amazon Redshift Spectrum.

Formats de données pour Redshift Spectrum

Redshift Spectrum prend en charge les formats de données structurées et semi-structurées suivants.

Format de fichier	Colonne	Prend en charge les lectures parallèles	Unité fractionnée
Parquet	Oui	Oui	Groupe de lignes
ORC	Oui	Oui	Stripe
RCFile	Oui	Oui	Groupe de lignes
TextFile	Non	Oui	Rangée
SequenceFile	Non	Oui	Ligne ou bloc
RegexSerde	Non	Oui	Rangée
OpenCSV	Non	Oui	Rangée
AVRO	Non	Oui	Bloc
Ion	Non	Non	N/A
JSON	Non	Non	N/A

Dans le tableau précédent, les titres indiquent ce qui suit :

- Colonnaire – Si le format de fichier stocke physiquement les données dans une structure orientée colonnes par opposition à une structure orientée lignes.

- Prise en charge des lectures parallèles – Indique si le format de fichier prend en charge la lecture de blocs individuels dans le fichier. La lecture de blocs individuels permet de répartir le traitement d'un fichier sur plusieurs requêtes Redshift Spectrum indépendantes, au lieu de devoir lire le fichier complet en une seule requête.
- Unité de fractionnement – Pour les formats de fichiers qui peuvent être lus en parallèle, l'unité de fractionnement est le plus petit morceau de données qu'une seule requête Redshift Spectrum peut traiter.

> **Note**
>
> Les valeurs d'horodatage dans les fichiers texte doivent être au format `yyyy-MM-dd HH:mm:ss.SSSSSS`, comme illustré par la valeur d'horodatage suivante : `2017-05-01 11:30:59.000000`.

Nous recommandons d'utiliser un format de fichier de stockage en colonnes, tel qu'Apache Parquet. Amazon Redshift prend en charge le format de fichier de données Apache Parquet v1. Avec un format de fichier de stockage en colonnes, vous pouvez minimiser le transfert de données hors d'Amazon S3 en ne sélectionnant que les colonnes dont vous avez besoin.

Types de compression pour Redshift Spectrum

Il est vivement recommandé de compresser vos fichiers de données afin de réduire l'espace de stockage, d'améliorer les performances et de diminuer les coûts. Redshift Spectrum reconnaît les types de compression de fichiers en fonction de l'extension de fichier.

Redshift Spectrum prend en charge les types de compression et extensions suivants.

Algorithme de compression	Extension de fichier	Prend en charge les lectures parallèles
Gzip	.gz	Non
Bzip2	.bz2	Oui
Snappy	.snappy	Non

Vous pouvez appliquer la compression à différents niveaux. Le plus souvent, vous compressez un fichier entier ou des blocs individuels dans un fichier. La compression des formats de colonnes au niveau du fichier n'entraîne pas d'avantages en termes de performances.

Pour que Redshift Spectrum puisse lire un fichier en parallèle, les conditions suivantes doivent être remplies :

- Le format de fichier prend en charge les lectures parallèles.
- La compression au niveau du fichier, le cas échéant, prend en charge les lectures parallèles.

Il importe peu que les unités de fractionnement individuelles d'un fichier soient compressées à l'aide d'un algorithme de compression pouvant être lu en parallèle, car chaque unité de fractionnement est traitée par une seule requête Redshift Spectrum. Les fichiers Parquet compressés par Snappy illustrent ce cas de figure. Les groupes de lignes individuels dans le fichier Parquet sont compressés à l'aide de Snappy, mais la structure de niveau supérieur du fichier reste non compressée. Dans ce cas, le fichier peut être lu en parallèle car chaque requête Redshift Spectrum peut lire et traiter des groupes de lignes individuels depuis Amazon S3.

Chiffrement pour Redshift Spectrum

Redshift Spectrum déchiffre de manière transparente les fichiers de données chiffrés à l'aide des options de chiffrement suivantes :

- Chiffrement côté serveur (SSE-S3) à l'aide d'une clé de chiffrement AES-256 gérée par Amazon S3.
- Chiffrement côté serveur avec des clés gérées par AWS Key Management Service (SSE-KMS).

Redshift Spectrum ne prend pas en charge le chiffrement côté client d'Amazon S3. Pour plus d'informations sur le chiffrement côté serveur, consultez Protection des données à l'aide du chiffrement côté serveur dans le Guide de l'utilisateur Amazon Simple Storage Service.

Amazon Redshift utilise le traitement massivement parallèle (MPP) pour obtenir une exécution rapide de requêtes complexes opérant sur de grandes quantités de données. Redshift Spectrum applique le même principe pour interroger les données externes, en utilisant si nécessaire plusieurs instances Redshift Spectrum pour analyser les fichiers. Placez les fichiers dans un dossier distinct pour chaque table.

Vous pouvez optimiser vos données pour un traitement parallèle en procédant comme suit :

- Si votre format de fichier ou votre compression ne prend pas en charge la lecture en parallèle, divisez les fichiers volumineux en plusieurs fichiers plus petits. Nous vous recommandons d'utiliser des fichiers d'une taille comprise entre 64 Mo et 1 Go.
- Faites en sorte que tous les fichiers aient à peu près la même taille. Redshift Spectrum ne peut pas distribuer équitablement la charge de travail si certains fichiers sont beaucoup plus volumineux que d'autres.

Schémas externes dans Amazon Redshift Spectrum

Cette rubrique explique comment créer et utiliser des schémas externes avec Redshift Spectrum. Les schémas externes sont des ensembles de tables que vous utilisez comme références pour accéder à des données en dehors de votre cluster Amazon Redshift. Ces tables contiennent des métadonnées relatives aux données externes lues par Redshift Spectrum.

Toutes les tables externes doivent être créées dans un schéma externe, que vous créez à l'aide d'une instruction CREATE EXTERNAL SCHEMA.

> **Note**
>
> Dans certaines applications, les termes base de données et schéma sont utilisés indifféremment. Dans Amazon Redshift, nous utilisons le terme schéma.

Un schéma externe Amazon Redshift fait référence à une base de données externe dans un catalogue de données externe. Vous pouvez créer la base de données externe dans Amazon Redshift, dans Amazon Athena, dans AWS Glue Data Catalog ou dans un métastore Apache Hive, tel qu'Amazon EMR. Si vous la créez une base de données externe dans Amazon Redshift, elle réside dans le catalogue de données Athena. Pour créer une base de données dans un metastore Hive, vous devez la créer dans votre application Hive.

Amazon Redshift a besoin d'une autorisation pour accéder au catalogue de données dans Athena et aux fichiers de données dans Amazon S3 en votre nom. Pour fournir cette autorisation, vous devez d'abord créer un rôle AWS Identity and Access Management (IAM). Ensuite, vous attachez le rôle à votre cluster et fournissez le Amazon Resource Name (ARN) pour le rôle dans la déclaration Amazon Redshift `CREATE EXTERNAL SCHEMA`. Pour de plus amples informations concernant l'autorisation, consultez Politiques IAM pour Amazon Redshift Spectrum.

Pour créer une base de données externe en même temps que vous créez un schéma externe, spécifiez `FROM DATA CATALOG` et incluez la clause `CREATE EXTERNAL DATABASE` dans l'instruction `CREATE EXTERNAL SCHEMA`.

L'exemple suivant permet de créer un schéma externe nommé `spectrum_schema` en utilisant la base de données externe `spectrum_db`.

```
create external schema spectrum_schema from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
create external database if not exists;
```

Si vous gérez votre catalogue de données à l'aide d'Athena, indiquez le nom de la base de données Athena et la région AWS dans laquelle se trouve le catalogue de données Athena.

L'exemple suivant crée un schéma externe en utilisant la base de données `sampledb` par défaut dans le catalogue de données Athena.

```
create external schema athena_schema from data catalog
database 'sampledb'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
region 'us-east-2';
```

> **Note**
>
> Le `region` paramètre fait référence à la AWS région dans laquelle se trouve le catalogue de données Athena, et non à l'emplacement des fichiers de données dans Amazon S3.

Si vous gérez votre catalogue de données à l'aide d'un métastore Hive, comme Amazon EMR, vos groupes de sécurité doivent être configurés pour autoriser le trafic entre les clusters.

Dans l'instruction CREATE EXTERNAL SCHEMA, spécifiez `FROM HIVE METASTORE` et incluez l'URI et le numéro de port du metastore. L'exemple suivant permet de créer un schéma externe en utilisant une base de données de metastore Hive nommée `hive_db`.

```
create external schema hive_schema
from hive metastore
database 'hive_db'
```

```
uri '172.10.10.10' port 99
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
```

Pour afficher les schémas externes correspondant à votre cluster, interrogez la table de catalogue PG_EXTERNAL_SCHEMA ou la vue SVV_EXTERNAL_SCHEMAS. L'exemple suivant interroge SVV_EXTERNAL_SCHEMAS qui joint PG_EXTERNAL_SCHEMA et PG_NAMESPACE.

```
select * from svv_external_schemas
```

Pour connaître la syntaxe complète de la commande et voir des exemples, consultez CREATE EXTERNAL SCHEMA.

Utiliser des catalogues externes dans Amazon Redshift Spectrum

Les métadonnées des bases de données externes et des tables externes d'Amazon Redshift Spectrum sont stockées dans un catalogue de données externes. Par défaut, les métadonnées de Redshift Spectrum sont stockées dans un catalogue de données Athena. Vous pouvez afficher et gérer les bases de données et les tables Redshift Spectrum dans votre console Athena.

Vous pouvez également créer et gérer des bases de données et des tables externes à l'aide du langage de définition de données (DDL) Hive en utilisant Athena ou un métastore Hive, tel que Amazon EMR.

> **Note**
>
> Nous vous recommandons d'utiliser Amazon Redshift pour créer et gérer des bases de données externes et des tables externes dans Redshift Spectrum.

Affichage des bases de données Redshift Spectrum dans Athena et AWS Glue

Vous pouvez créer une base de données externe en incluant la clause CREATE EXTERNAL DATABASE IF NOT EXISTS dans votre instruction CREATE EXTERNAL SCHEMA. Le cas échéant, les métadonnées de la base de données externe sont stockées dans votre catalogue de données. Tout comme les métadonnées relatives aux tables externes que vous créez qui sont qualifiées par le schéma externe sont également stockées dans votre catalogue de données .

Athena et AWS Glue maintenez un catalogue de données pour chaque support pris en charge. Région AWS Pour consulter les métadonnées d'une table, connectez-vous à Athena ou AWS Glue à

la console. Dans Athena, choisissez Sources de données, votre AWS Glue, puis consultez les détails de votre base de données. Dans AWS Glue, choisissez Bases de données, votre base de données externe, puis affichez les détails de votre base de données.

Si vous créez et gérez vos tables externes à l'aide d'Athena, enregistrez la base de données en utilisant CREATE EXTERNAL SCHEMA. Par exemple, la commande suivante enregistre la base de données Athena nommée `sampledb`.

```
create external schema athena_sample
from data catalog
database 'sampledb'
iam_role 'arn:aws:iam::123456789012:role/mySpectrumRole'
region 'us-east-1';
```

Lorsque vous interrogez la vue système SVV_EXTERNAL_TABLES, vous voyez les tables dans la base de données Athena `sampledb` et également celles que vous avez créées dans Amazon Redshift.

```
select * from svv_external_tables;
```

```
schemaname    | tablename        | location

--------------+------------------
+-----------------------------------------------------
athena_sample | elb_logs         | s3://athena-examples/elb/plaintext
athena_sample | lineitem_1t_csv  | s3://myspectrum/tpch/1000/lineitem_csv

athena_sample | lineitem_1t_part | s3://myspectrum/tpch/1000/lineitem_partition

spectrum      | sales            | s3://redshift-downloads/tickit/spectrum/sales

spectrum      | sales_part       | s3://redshift-downloads/tickit/spectrum/sales_part
```

Enregistrement d'une base de données de metastore Apache Hive

Si vous créez des tables externes dans un metastore Apache Hive, vous pouvez les enregistrer dans Redshift Spectrum à l'aide de l'instruction CREATE EXTERNAL SCHEMA.

Dans l'instruction CREATE EXTERNAL SCHEMA, spécifiez la clause FROM HIVE METASTORE, ainsi que l'URI et le numéro de port du metastore Hive. Le rôle IAM doit inclure l'autorisation

d'accéder à Amazon S3 mais ne nécessite aucune autorisation Athena. L'exemple suivant enregistre un metastore Hive.

```
create external schema if not exists hive_schema
from hive metastore
database 'hive_database'
uri 'ip-10-0-111-111.us-west-2.compute.internal' port 9083
iam_role 'arn:aws:iam::123456789012:role/mySpectrumRole';
```

Permettre à votre cluster Amazon Redshift d'accéder à votre cluster Amazon EMR

Si votre métastore Hive se trouve dans Amazon EMR, vous devez donner à votre cluster Amazon Redshift l'accès à votre cluster Amazon EMR. Pour ce faire, vous devez créer un groupe EC2 de sécurité Amazon. Vous autorisez ensuite tout le trafic entrant vers le groupe de EC2 sécurité depuis le groupe de sécurité de votre cluster Amazon Redshift et le groupe de sécurité de votre cluster Amazon EMR. Vous ajoutez ensuite la EC2 sécurité à votre cluster Amazon Redshift et à votre cluster Amazon EMR.

Afficher le nom du groupe de sécurité du cluster Amazon Redshift

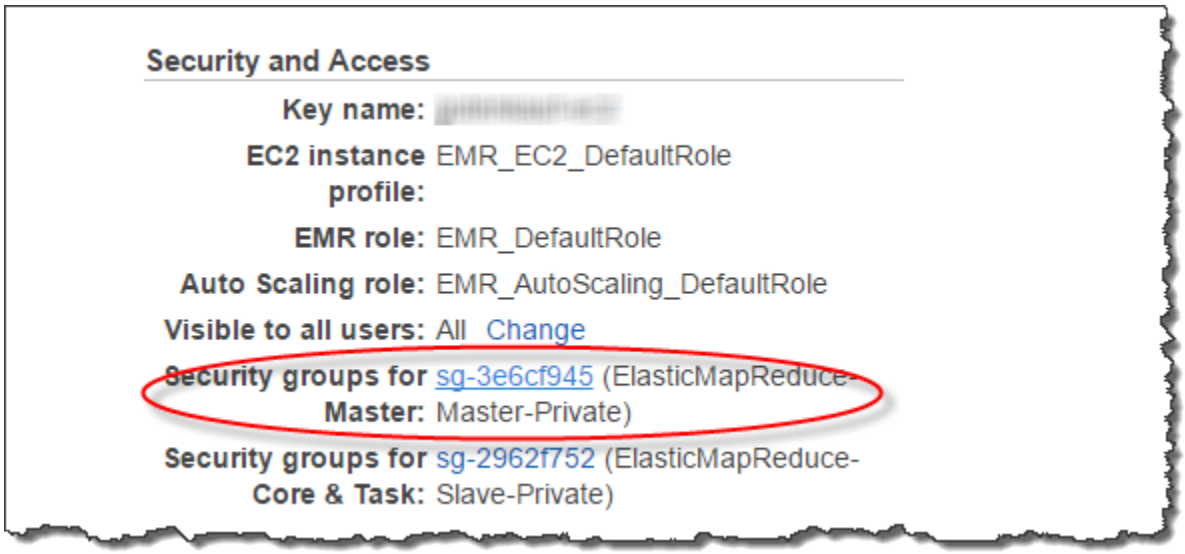
Pour afficher un groupe de sécurité, procédez comme suit :

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez le cluster dans la liste pour ouvrir ses détails.
3. Choisissez Properties (Propriétés) et affichez la section Network and security settings (Paramètres de réseau et de sécurité).
4. Recherchez votre groupe de sécurité dans VPC security group (Groupe de sécurité VPC) et prenez-en note.

Afficher le nom du groupe de sécurité du nœud principal Amazon EMR

1. Ouvrez votre cluster Amazon EMR. Pour plus d'informations, consultez Utiliser des configurations de sécurité pour configurer la sécurité du cluster dans le Guide de gestion Amazon EMR.

2. Sous Security and access (Sécurité et accès), notez le nom du groupe de sécurité du nœud principal Amazon EMR.



Pour créer ou modifier un groupe de EC2 sécurité Amazon afin d'autoriser la connexion entre Amazon Redshift et Amazon EMR

1. Dans le EC2 tableau de bord Amazon, choisissez Security groups. Pour plus d'informations, consultez la section Règles relatives aux groupes de sécurité dans le guide de EC2 l'utilisateur Amazon
2. Sélectionnez Create security group (Créer un groupe de sécurité).
3. Si vous utilisez VPC, choisissez le VPC dans lequel se trouvent vos clusters Amazon Redshift et Amazon EMR.
4. Ajoutez une règle entrante.
    1. Pour Type, choisissez Custom TCP (TCP personnalisé).
    2. Pour Source, choisissez Personnalisé.
    3. Entrez le nom de votre groupe de sécurité Amazon Redshift.
5. Ajoutez une autre règle entrante.
    1. Pour Type, choisissez TCP.
    2. Pour Plage de ports, entrez 9083.

> **Note**
>
> Le port par défaut pour un HMS EMR est 9083. Si votre HMS utilise un port distinct, spécifiez ce port dans la règle entrante et dans la définition de schéma externe.

3. Pour Source, choisissez Personnalisé.

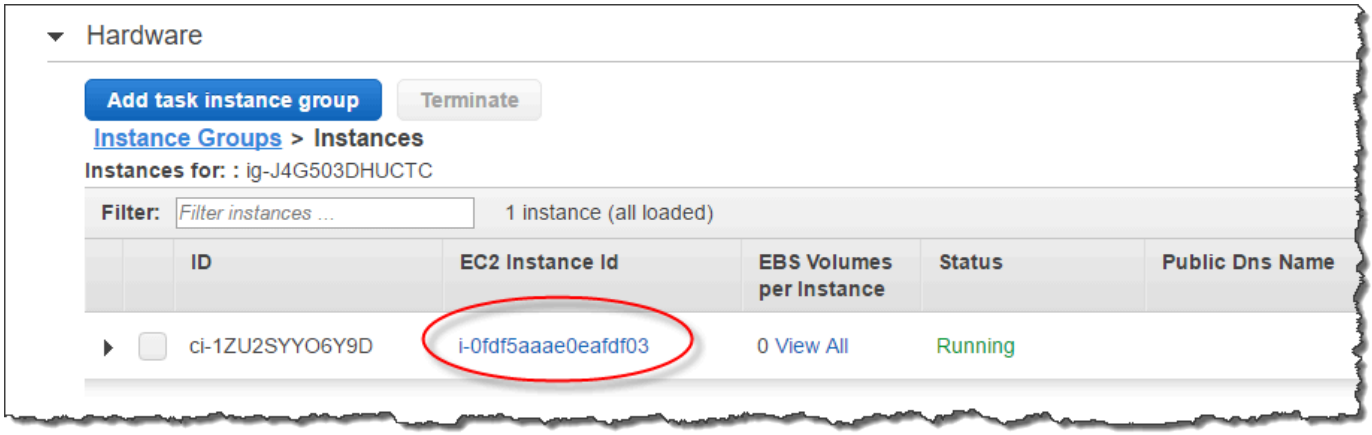
6. Saisissez un nom et une description pour le groupe de sécurité.
7. Sélectionnez Create security group (Créer un groupe de sécurité).

Pour ajouter le groupe EC2 de sécurité Amazon que vous avez créé lors de la procédure précédente à votre cluster Amazon Redshift

1. Dans Amazon Redshift, choisissez votre cluster.
2. Choisissez Propriétés.
3. Affichez les paramètres réseau et de sécurité et choisissez Edit (Modifier).
4. Dans VPC security group (Groupe de sécurité VPC), choisissez le nouveau nom du groupe de sécurité.
5. Sélectionnez Enregistrer les modifications.

Pour ajouter le groupe EC2 de sécurité Amazon à votre cluster Amazon EMR

1. Dans Amazon EMR, choisissez votre cluster. Pour plus d'informations, consultez Utiliser des configurations de sécurité pour configurer la sécurité du cluster dans le Guide de gestion Amazon EMR.
2. Sous Hardware (Matériel), choisissez le lien correspondant au nœud principal.
3. Choisissez le lien dans la colonne ID de l'EC2 instance.



4. Choisissez Actions, Security (Sécurité), Change security groups (Modifier les groupes de sécurité).
5. Dans Associated sercurity groups (Groupes de sécurité associés), choisissez le nouveau groupe de sécurité, puis choisissez Add security group (Ajouter un groupe de sécurité).
6. Choisissez Enregistrer.

Tables externes pour Redshift Spectrum

Cette rubrique explique comment créer et utiliser des tables externes avec Redshift Spectrum. Les tables externes sont des tables que vous utilisez comme références pour accéder à des données en dehors de votre cluster Amazon Redshift. Ces tables contiennent des métadonnées relatives aux données externes lues par Redshift Spectrum.

Vous créez une table externe dans un schéma externe. Pour créer des tables externes, vous devez être le propriétaire du schéma externe ou un superutilisateur. Pour transférer la propriété d'un schéma externe, utilisez ALTER SCHEMA pour modifier le propriétaire. L'exemple suivant remplace le propriétaire du schéma `spectrum_schema` par `newowner`.

```
alter schema spectrum_schema owner to newowner;
```

Pour exécuter une requête Redshift Spectrum, vous devez avoir les autorisations suivantes :

- Autorisations d'utilisation du schéma
- Autorisation de créer des tables temporaires dans la base de données actuelle

L'exemple suivant accorde l'autorisation d'utiliser le schéma `spectrum_schema` au groupe d'utilisateurs `spectrumusers`.

```
grant usage on schema spectrum_schema to group spectrumusers;
```

L'exemple suivant accorde une autorisation temporaire concernant la base de données `spectrumdb` au groupe d'utilisateurs `spectrumusers`.

```
grant temp on database spectrumdb to group spectrumusers;
```

Vous pouvez créer une table externe dans Amazon Redshift AWS Glue, Amazon Athena ou dans un métastore Apache Hive. Pour plus d'informations, consultez Premiers pas avec AWS Glue dans le guide du développeur AWS Glue , Démarrez dans le guide de l'utilisateur d'Amazon Athena, ou Apache Hive dans le guide du développeur Amazon EMR.

Si votre table externe est définie dans Athena ou dans AWS Glue un métastore Hive, vous devez d'abord créer un schéma externe qui fait référence à la base de données externe. Vous pouvez alors faire référence à la table externe dans votre instruction SELECT en faisant précéder le nom de la table du nom du schéma, sans avoir à créer la table dans Amazon Redshift. Pour de plus amples informations, veuillez consulter Schémas externes dans Amazon Redshift Spectrum.

Pour permettre à Amazon Redshift d'afficher les tables dans le AWS Glue Data Catalog, ajoutez-les `glue:GetTable` au rôle Amazon Redshift IAM. Sinon, vous risquez de recevoir une erreur similaire à ce qui suit.

```
RedshiftIamRoleSession is not authorized to perform: glue:GetTable on resource: *;
```

Par exemple, supposons que vous ayez une table externe nommée `lineitem_athena` définie dans un catalogue externe Athena. Vous pouvez dans ce cas définir un schéma externe nommé `athena_schema`, puis interroger la table à l'aide de l'instruction SELECT suivante.

```
select count(*) from athena_schema.lineitem_athena;
```

Pour définir une table externe dans Amazon Redshift, utilisez la commande CREATE EXTERNAL TABLE. L'instruction de la table externe définit les colonnes de la table, le format des fichiers de données ainsi que l'emplacement des données dans Amazon S3. Redshift Spectrum analyse les fichiers dans le dossier spécifié, mais pas dans les sous-dossiers. Redshift Spectrum ignore les

fichiers masqués ainsi que les fichiers dont le nom commence par un point, un trait de soulignement ou une marque de hachage (. , _ ou #) ou se termine par un tilde (~).

L'exemple suivant crée une table nommée SALES dans le schéma externe Amazon Redshift nommé `spectrum`. Les données figurent dans des fichiers texte délimités par des tabulations.

```
create external table spectrum.sales(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales/'
table properties ('numRows'='172000');
```

Pour afficher les tables externes, interrogez la vue système SVV_EXTERNAL_TABLES.

Pseudocolonnes

Par défaut, Amazon Redshift crée des tables externes avec les pseudo-colonnes `$path`, `$size` et `$spectrum_oid`. Sélectionnez la colonne `$path` pour afficher le chemin d'accès aux fichiers de données sur Amazon S3 et sélectionnez la colonne `$size` pour afficher la taille des données de chaque ligne renvoyée par une requête. La colonne `$spectrum_oid` permet d'effectuer des requêtes corrélées avec Redshift Spectrum. Pour obtenir un exemple, consultez Exemple : exécution de sous-requêtes corrélées dans Redshift Spectrum. Vous devez délimiter les noms de colonne `$path`, `$size` et `$spectrum_oid` par des guillemets doubles. Une clause SELECT * ne renvoie pas les pseudo-colonnes. Vous devez inclure explicitement les noms de colonne `$path`, `$size` et `$spectrum_oid` dans votre requête, comme l'illustre l'exemple suivant.

```
select "$path", "$size", "$spectrum_oid"
from spectrum.sales_part where saledate = '2008-12-01';
```

Vous pouvez désactiver la création de pseudo-colonnes d'une séance en définissant le paramètre de configuration `spectrum_enable_pseudo_columns` avec la valeur `false`. Pour de plus amples informations, veuillez consulter spectrum_enable_pseudo_columns. Vous pouvez aussi désactiver uniquement la pseudo-colonne `$spectrum_oid` en définissant `enable_spectrum_oid` sur `false`. Pour de plus amples informations, veuillez consulter enable_spectrum_oid. Toutefois, la désactivation de la pseudo-colonne `$spectrum_oid` désactive également la prise en charge des requêtes corrélées avec Redshift Spectrum.

> **Important**
>
> La sélection de `$size`, `$path` ou `$spectrum_oid` entraîne des frais, car Redshift Spectrum analyse les fichiers de données sur Amazon S3 pour déterminer la taille de l'ensemble de résultats. Pour plus d'informations, veuillez consulter la rubrique Tarification d'Amazon Redshift.

Exemple de pseudocolonnes

L'exemple suivant renvoie la taille totale des fichiers de données associés pour une table externe.

```
select distinct "$path", "$size"
from spectrum.sales_part;

 $path                                                                      | $size
----------------------------------------------------------------------------+-------
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/ |  1616
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/ |  1444
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/ |  1644
```

Partitionnement des tables externes Redshift Spectrum

Lorsque vous partitionnez vos données, vous pouvez restreindre la quantité de données analysées par Redshift Spectrum en les filtrant en fonction de n'importe quelle clé de partition.

Il est courant de les partitionner selon des critères temporels. Par exemple, vous pouvez les partitionner en fonction de l'année, du mois, de la date et de l'heure. Si les données sont issues de plusieurs sources, vous pouvez les partitionner selon un identificateur de source de données et une date.

La procédure suivante décrit comment partitionner les données.

Pour partitionner les données

1. Stockez les données dans des dossiers d'Amazon S3 en fonction de la clé de partition.

   Créez pour chaque valeur de partition un dossier que vous nommerez à l'aide de la clé et de la valeur de partition. Par exemple, si vous partitionnez les données par date, vos dossiers peuvent être nommés `saledate=2017-04-01`, `saledate=2017-04-02`, etc. Redshift Spectrum analyse les fichiers dans le dossier de partition et tous les sous-dossiers. Redshift Spectrum ignore les fichiers masqués ainsi que les fichiers dont le nom commence par un point, un trait de soulignement ou une marque de hachage (. , _ ou #) ou se termine par un tilde (~).

2. Créez une table externe, puis spécifiez la clé de partition dans la clause PARTITIONED BY.

   La clé de partition ne peut pas correspondre au nom d'une colonne de la table. Le type de données peut être SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE ou TIMESTAMP.

3. Ajoutez des partitions.

   En utilisant ALTER TABLE … ADD PARTITION, ajoutez chaque partition, en spécifiant la colonne de la partition et la valeur de la clé, ainsi que l'emplacement du dossier de partition dans Amazon S3. Vous pouvez ajouter plusieurs partitions en une seule instruction ALTER TABLE … ADD. L'exemple qui suit ajoute des partitions pour `'2008-01'` et `'2008-03'`.

```
alter table spectrum.sales_part add
partition(saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/
saledate=2008-01/'
partition(saledate='2008-03-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/
saledate=2008-03/';
```

> **Note**
>
> Si vous utilisez le AWS Glue catalogue, vous pouvez ajouter jusqu'à 100 partitions à l'aide d'une seule instruction ALTER TABLE.

Exemples de partitionnement de données

Dans cet exemple, vous devez créer une table externe partitionnée par une seule clé de partition et une table externe partitionnée par deux clés de partition.

Les exemples de données de cet exemple se trouvent dans un compartiment Amazon S3 qui donne un accès en lecture à tous les AWS utilisateurs authentifiés. Votre cluster et vos fichiers de données externes doivent se trouver dans la même Région AWS. L'exemple de compartiment de données se trouve dans la région USA Est (Virginie du Nord) (us-east-1). Pour pouvoir accéder aux données à l'aide de Redshift Spectrum, votre cluster doit donc également se trouver dans la région us-east-1. Pour répertorier les dossiers dans Amazon S3, exécutez la commande suivante.

```
aws s3 ls s3://redshift-downloads/tickit/spectrum/sales_partition/
```

```
PRE saledate=2008-01/
PRE saledate=2008-03/
PRE saledate=2008-04/
PRE saledate=2008-05/
PRE saledate=2008-06/
PRE saledate=2008-12/
```

Si vous ne possédez pas encore de schéma externe, exécutez la commande ci-dessous. Remplacez votre rôle (IAM) par le nom de ressource Amazon AWS Identity and Access Management (ARN).

```
create external schema spectrum
from data catalog
database 'spectrumdb'
iam_role 'arn:aws:iam::123456789012:role/myspectrumrole'
create external database if not exists;
```

Exemple 1 : Partitionnement avec une seule clé de partition

Dans l'exemple suivant, vous devez créer une table externe partitionnée par mois.

Pour créer une table externe partitionnée par mois, exécutez la commande suivante.

```
create external table spectrum.sales_part(
salesid integer,
listid integer,
```

```
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
partitioned by (saledate char(10))
row format delimited
fields terminated by '|'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales_partition/'
table properties ('numRows'='172000');
```

Exécutez la commande ALTER TABLE suivante pour ajouter les partitions.

```
alter table spectrum.sales_part add
partition(saledate='2008-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/'

partition(saledate='2008-03')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/'

partition(saledate='2008-04')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-04/';
```

Pour sélectionner les données de la table partitionnée, exécutez la requête suivante.

```
select top 5 spectrum.sales_part.eventid, sum(spectrum.sales_part.pricepaid)
from spectrum.sales_part, event
where spectrum.sales_part.eventid = event.eventid
  and spectrum.sales_part.pricepaid > 30
  and saledate = '2008-01'
group by spectrum.sales_part.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
   4124 | 21179.00
   1924 | 20569.00
   2294 | 18830.00
```

```
   2260 | 17669.00
   6032 | 17265.00
```

Pour afficher les partitions de la table externe, interrogez la vue système SVV_EXTERNAL_PARTITIONS.

```
select schemaname, tablename, values, location from svv_external_partitions
where tablename = 'sales_part';
```

```
schemaname | tablename  | values      | location

-----------+------------+-------------
+-------------------------------------------------------------------------
spectrum   | sales_part | ["2008-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-01
spectrum   | sales_part | ["2008-03"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-03
spectrum   | sales_part | ["2008-04"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-04
```

Exemple 2 : Partitionnement avec plusieurs clés de partition

Pour créer une table externe partitionnée par `date` et `eventid`, exécutez la commande suivante.

```
create external table spectrum.sales_event(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
partitioned by (salesmonth char(10), event integer)
row format delimited
fields terminated by '|'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/salesevent/'
table properties ('numRows'='172000');
```

Exécutez la commande ALTER TABLE suivante pour ajouter les partitions.

```
alter table spectrum.sales_event add
partition(salesmonth='2008-01', event='101')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-01/
event=101/'

partition(salesmonth='2008-01', event='102')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-01/
event=102/'

partition(salesmonth='2008-01', event='103')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-01/
event=103/'

partition(salesmonth='2008-02', event='101')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-02/
event=101/'

partition(salesmonth='2008-02', event='102')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-02/
event=102/'

partition(salesmonth='2008-02', event='103')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-02/
event=103/'

partition(salesmonth='2008-03', event='101')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-03/
event=101/'

partition(salesmonth='2008-03', event='102')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-03/
event=102/'

partition(salesmonth='2008-03', event='103')
location 's3://redshift-downloads/tickit/spectrum/salesevent/salesmonth=2008-03/
event=103/';
```

Exécutez la requête suivante pour sélectionner les données de la table partitionnée.

```
select spectrum.sales_event.salesmonth, event.eventname,
 sum(spectrum.sales_event.pricepaid)
```

```
from spectrum.sales_event, event
where spectrum.sales_event.eventid = event.eventid
  and salesmonth = '2008-02'
 and (event = '101'
 or event = '102'
 or event = '103')
group by event.eventname, spectrum.sales_event.salesmonth
order by 3 desc;
```

```
salesmonth | eventname       | sum
-----------+-----------------+--------
2008-02    | The Magic Flute | 5062.00
2008-02    | La Sonnambula   | 3498.00
2008-02    | Die Walkure     |  534.00
```

Mappage de colonnes de table externe à des colonnes ORC

Vous utilisez les tables externes Amazon Redshift Spectrum pour interroger les données des fichiers au format ORC. Le format Optimized Row Columnar (ORC) est un format de fichier de stockage en colonnes qui prend en charge les structures de données imbriquées. Pour plus d'informations sur l'interrogation de données imbriquées, consultez Interrogation de données imbriquées avec Amazon Redshift Spectrum.

Lorsque vous créez une table externe qui fait référence à des données dans un fichier ORC, vous mappez chaque colonne de la table externe à une colonne dans les données ORC. Pour ce faire, vous utilisez l'une des méthodes suivantes :

- Mappage par position
- Mappage par nom de colonne

Le mappage par nom de colonne est l'option par défaut.

Mappage par position

Avec le mappage par position, la première colonne définie dans la table externe est mappée à la première colonne du fichier de données ORC, la deuxième colonne à la deuxième colonne, et ainsi de suite. Pour le mappage par position, l'ordre des colonnes dans la table externe et dans le fichier ORC doivent correspondre. Si l'ordre des colonnes ne correspond pas, vous pouvez mapper les colonnes par nom.

> **Important**
>
> Dans des versions précédentes, Redshift Spectrum utilisait le mappage par position par défaut. Si vous avez besoin de continuer à utiliser le mappage par position pour des tables existantes, définissez la propriété de table `orc.schema.resolution` sur `position`, comme illustré dans l'exemple suivant.
>
> ```
> alter table spectrum.orc_example
> set table properties('orc.schema.resolution'='position');
> ```

Par exemple, la table `SPECTRUM.ORC_EXAMPLE` est définie comme suit.

```
create external table spectrum.orc_example(
int_col int,
float_col float,
nested_col struct<
  "int_col" : int,
  "map_col" : map<int, array<float >>
   >
) stored as orc
location 's3://example/orc/files/';
```

La structure de table peut être extraite comme suit.

```
• 'int_col' : int
• 'float_col' : float
• 'nested_col' : struct
   o 'int_col' : int
   o 'map_col' : map
      - key : int
      - value : array
         - value : float
```

Le fichier ORC sous-jacent a la structure de fichier suivante.

```
• ORC file root(id = 0)
   o 'int_col' : int (id = 1)
   o 'float_col' : float (id = 2)
   o 'nested_col' : struct (id = 3)
```

```
      - 'int_col' : int (id = 4)
      - 'map_col' : map (id = 5)
         - key : int (id = 6)
         - value : array (id = 7)
            - value : float (id = 8)
```

Dans cet exemple, vous pouvez mapper chaque colonne de la table externe à une colonne du fichier ORC strictement par position. Voici une illustration du mappage.

Nom de colonne de la table externe	ID de colonne ORC	Nom de colonne ORC
int_col	1	int_col
float_col	2	float_col
nested_col	3	nested_col
nested_col.int_col	4	int_col
nested_col.map_col	5	map_col
nested_col.map_col.key	6	NA
nested_col.map_col.value	7	NA
nested_col.map_col.value.item	8	NA

Mappage par nom de colonne

À l'aide du mappage par nom, vous mappez des colonnes d'une table externe à des colonnes nommées de fichiers ORC sur le même niveau avec le même nom.

Par exemple, supposons que vous souhaitez mapper la table de l'exemple précédent, `SPECTRUM.ORC_EXAMPLE`, avec un fichier ORC qui utilise la structure de fichier suivante.

```
• ORC file root(id = 0)
   o 'nested_col' : struct (id = 1)
      - 'map_col' : map (id = 2)
         - key : int (id = 3)
         - value : array (id = 4)
```

```
            - value : float (id = 5)
      - 'int_col' : int (id = 6)
   o 'int_col' : int (id = 7)
   o 'float_col' : float (id = 8)
```

À l'aide du mappage par position, Redshift Spectrum tente le mappage suivant.

Nom de colonne de la table externe	ID de colonne ORC	Nom de colonne ORC
int_col	1	struct
float_col	7	int_col
nested_col	8	float_col

Lorsque vous interrogez une table avec le mappage par position précédent, la commande SELECT échoue pour la validation du type parce que les structures sont différentes.

Vous pouvez mapper la même table externe aux deux structures de fichier illustrés dans les exemples précédents à l'aide du mappage par nom de colonne. Les colonnes de table `int_col`, `float_col` et `nested_col` sont mappées par nom de colonne aux colonnes avec les mêmes noms dans le fichier ORC. La colonne de table nommée `nested_col`dans la table externe est une colonne `struct` avec des sous-colonnes nommées `map_col` et `int_col`. Les sous-colonnes sont mappées correctement aux colonnes correspondantes dans le fichier ORC par nom de colonne.

Créer des tables externes pour les données gérées dans Apache Hudi

Pour interroger des données au format Apache Hudi Copy On Write (CoW), vous pouvez utiliser les tables externes Amazon Redshift Spectrum. Une table Hudi Copy On Write est une collection de fichiers Apache Parquet stockés dans Amazon S3. Vous pouvez lire les tables Copy On Write (Copie sur écriture, CoW) qui sont créées et modifiées avec les opérations d'écriture insert, delete et upsert dans Apache Hudi versions 0.5.2, 0.6.0, 0.7.0, 0.8.0, 0.9.0, 0.10.0, 0.10.1, 0.11.0 et 0.11.1. Par exemple, les tables d'amorçage ne sont pas prises en charge. Pour de plus amples informations, consultez Table Copy On Write dans la documentation open source Apache Hudi.

Lorsque vous créez une table externe qui fait référence à des données au format CoW Hudi, vous mappez chaque colonne de la table externe à une colonne des données Hudi. Le mappage se fait par colonne.

Les instructions DDL (Data Definition Language) pour les tables Hudi partitionnées et non partitionnées sont similaires à celles des autres formats de fichier Apache Parquet. Pour les tables Hudi, vous définissez `INPUTFORMAT` sur `org.apache.hudi.hadoop.HoodieParquetInputFormat`. Le paramètre `LOCATION` doit être tourné vers le dossier de base de la table Hudi qui contient le dossier `.hoodie`, qui est nécessaire pour établir la chronologie de validation Hudi. Dans certains cas, une opération SELECT sur une table Hudi peut échouer avec le message No valid Hudi commit timeline found (Aucune chronologie de validation Hudi valide n'a été trouvée). Si c'est le cas, vérifiez si le dossier `.hoodie` est à l'emplacement correct et contient une chronologie de validation Hudi valide.

> **Note**
>
> Le format Apache Hudi n'est pris en charge que lorsque vous utilisez un AWS Glue Data Catalog. Il n'est pas pris en charge lorsque vous utilisez un métastore Apache Hive comme catalogue externe.

Le format DDL pour définir une table non partitionnée est au format suivant.

```
CREATE EXTERNAL TABLE tbl_name (columns)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hudi.hadoop.HoodieParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://s3-bucket/prefix'
```

Le DDL pour définir une table partitionnée est au format suivant.

```
CREATE EXTERNAL TABLE tbl_name (columns)
PARTITIONED BY(pcolumn1 pcolumn1-type[,...])
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hudi.hadoop.HoodieParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://s3-bucket/prefix'
```

Pour ajouter des partitions à une table Hudi partitionnée, exécutez une commande ALTER TABLE ADD PARTITION dans laquelle le paramètre `LOCATION` est tourné vers le sous-dossier Amazon S3 avec les fichiers qui appartiennent à la partition.

Le DDL pour ajouter des partitions est au format suivant.

```
ALTER TABLE tbl_name
ADD IF NOT EXISTS PARTITION(pcolumn1=pvalue1[,...])
LOCATION 's3://s3-bucket/prefix/partition-path'
```

Créer des tables externes pour les données gérées dans Delta Lake

Pour interroger des données dans des tables Delta Lake, vous pouvez utiliser des tables externes Amazon Redshift Spectrum.

Pour accéder à une table Delta Lake à partir de Redshift Spectrum, générez un manifeste avant la requête. Un manifeste Delta Lake contient une liste des fichiers qui constituent un instantané cohérent de la table Delta Lake. Dans une table partitionnée, il y a un manifeste par partition. Une table Delta Lake est une collection de fichiers Apache Parquet stockés dans Amazon S3. Pour plus d'informations, consultez Delta Lake dans la documentation open source de Delta Lake.

Lorsque vous créez une table externe qui fait référence aux données des tables Delta Lake, vous mappez chaque colonne de la table externe à une colonne de la table Delta Lake. Le mappage se fait par nom de colonne.

La DDL pour les tables Delta Lake partitionnées et non partitionnées est similaire à celle des autres formats de fichiers Apache Parquet. Pour les tables Delta Lake, vous définissez `INPUTFORMAT` sur `org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat` et `OUTPUTFORMAT` sur `org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat`. Le paramètre `LOCATION` doit être tourné vers le dossier manifeste dans le dossier de base de la table. Si une opération SELECT sur une table Delta Lake échoue, pour des raisons possibles, consultez Limites et dépannage pour les tables Delta Lake.

Le format DDL pour définir une table non partitionnée est au format suivant.

```
CREATE EXTERNAL TABLE tbl_name (columns)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://s3-bucket/prefix/_symlink_format_manifest'
```

Le DDL pour définir une table partitionnée est au format suivant.

```
CREATE EXTERNAL TABLE tbl_name (columns)
PARTITIONED BY(pcolumn1 pcolumn1-type[,...])
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS
INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://s3-bucket>/prefix/_symlink_format_manifest'
```

Pour ajouter des partitions à une table Delta Lake partitionnée, exécutez une commande ALTER TABLE ADD PARTITION dans laquelle le paramètre LOCATION est tourné vers le sous-dossier Amazon S3 qui contient le manifeste de la partition.

Le DDL pour ajouter des partitions est au format suivant.

```
ALTER TABLE tbl_name
ADD IF NOT EXISTS PARTITION(pcolumn1=pvalue1[,...])
LOCATION
's3://s3-bucket/prefix/_symlink_format_manifest/partition-path'
```

Sinon, exécutez DDL qui est tourné directement vers le fichier manifeste Delta Lake.

```
ALTER TABLE tbl_name
ADD IF NOT EXISTS PARTITION(pcolumn1=pvalue1[,...])
LOCATION
's3://s3-bucket/prefix/_symlink_format_manifest/partition-path/manifest'
```

Limites et dépannage pour les tables Delta Lake

Tenez compte des éléments suivants lorsque vous interrogez des tables Delta Lake à partir de Redshift Spectrum :

- Si un manifeste est tourné vers un instantané ou une partition qui n'existe plus, les requêtes échouent jusqu'à ce qu'un nouveau manifeste valide ait été généré. Par exemple, cela peut résulter d'une opération VACUUM sur la table sous-jacente,
- Les manifestes Delta Lake fournissent uniquement une cohérence au niveau des partitions.

Le tableau suivant explique les raisons potentielles de certaines erreurs lorsque vous interrogez une table Delta Lake.

Error message (Message d'erreur)	Raison possible
Le manifeste Delta Lake dans le compartiment s3-bucket-1 ne peut pas contenir d'entrées dans le compartiment s3-bucket-2.	Les entrées de manifeste sont tournées vers des fichiers dans un compartiment Amazon S3 différent de celui spécifié.
Les fichiers Delta Lake doivent se trouver dans le même dossier.	Les entrées de manifeste sont tournées vers des fichiers qui ont un préfixe Amazon S3 différent de celui spécifié.
Le fichier filename figurant dans le manifeste Delta Lake manifest-path n'a pas été trouvé.	Un fichier répertorié dans le manifeste n'a pas été trouvé dans Amazon S3.
Erreur lors de la récupération du manifeste Delta Lake.	Le manifeste n'a pas été trouvé dans Amazon S3.
Chemin d'accès S3 non valide.	Une entrée dans le fichier manifeste n'est pas un chemin d'accès Amazon S3 valide ou le fichier manifeste a été corrompu.

Utilisation de tables Apache Iceberg avec Amazon Redshift

Cette rubrique explique comment utiliser des tables au format Apache Iceberg avec Redshift Spectrum ou Redshift Serverless. Apache Iceberg est un format très performant pour les grandes tables analytiques.

Vous pouvez utiliser Redshift Spectrum ou Redshift sans serveur pour interroger des tables Apache Iceberg cataloguées dans le AWS Glue Data Catalog. Apache Iceberg est un format de table open source pour les lacs de données. Pour en savoir plus, consultez Apache Iceberg dans la documentation Apache Iceberg.

Amazon Redshift assure une cohérence transactionnelle dans le cadre de l'interrogation des tables Apache Iceberg. Vous pouvez manipuler les données de vos tables à l'aide de services conformes aux propriétés ACID (atomicité, cohérence, isolation, durabilité) comme Amazon Athena et Amazon EMR tout en exécutant des requêtes avec Amazon Redshift. Amazon Redshift peut utiliser les statistiques de tables stockées dans les métadonnées Apache Iceberg pour optimiser les plans de requêtes et limiter les analyses de fichiers pendant le traitement des requêtes. Avec Amazon Redshift SQL, vous pouvez joindre des tables Redshift à des tables de lac de données.

Pour commencer à utiliser des tables Iceberg avec Amazon Redshift :

1. Créez une table Apache Iceberg sur une AWS Glue Data Catalog base de données à l'aide d'un service compatible tel qu'Amazon Athena ou Amazon EMR. Pour créer une table Iceberg à l'aide d'Athena, consultez Utilisation de tables Apache Iceberg dans le Guide de l'utilisateur Amazon Athena.
2. Créez un cluster Amazon Redshift ou un groupe de travail Redshift sans serveur avec un rôle IAM associé qui permet d'accéder à votre lac de données. Pour plus d'informations sur la création de clusters ou de groupes de travail, consultez Commencer avec les entrepôts de données provisionnés Amazon Redshift et Commencer avec les entrepôts de données sans serveur Redshift dans le guide de démarrage Amazon Redshift.
3. Connectez-vous à votre cluster ou groupe de travail à l'aide de l'éditeur de requêtes v2 ou d'un client SQL tiers. Pour plus d'informations sur la façon de se connecter à l'aide de l'éditeur de requêtes v2, consultez la section Connexion à un entrepôt de données Amazon Redshift à l'aide des outils client SQL dans le guide de gestion Amazon Redshift.
4. Créez un schéma externe dans votre base de données Amazon Redshift pour une base de données de catalogue de données spécifique qui comprend vos tables Iceberg. Pour obtenir des informations sur la création d'un schéma externe, consultez Schémas externes dans Amazon Redshift Spectrum.
5. Exécutez des requêtes SQL pour accéder aux tables Iceberg dans le schéma externe que vous avez créé.

Éléments à prendre en considération lors de l'utilisation de tables Apache Iceberg avec Amazon Redshift

Tenez compte des points suivants lorsque vous utilisez Amazon Redshift avec des tables Iceberg :

- Prise en charge des versions d'Iceberg – Amazon Redshift prend en charge l'exécution de requêtes sur les versions suivantes de tables Iceberg :
  - La version 1 définit la façon dont les grandes tables analytiques sont gérées en utilisant des fichiers de données immuables.
  - La version 2 ajoute la possibilité de prendre en charge les mises à jour et les suppressions de niveau ligne tout en laissant les fichiers de données existants inchangés et en gérant les modifications de données de tables à l'aide de fichiers de suppression.

Pour connaître la différence entre les tables de version 1 et de version 2, consultez Format version changes dans la documentation Apache Iceberg.

- Requêtes uniquement – Amazon Redshift prend en charge l'accès en lecture seule aux tables Apache Iceberg. Il prend en charge les requêtes de sélection transactionnelles cohérentes. Vous pouvez utiliser un service comme Amazon Athena pour définir et mettre à jour le schéma des tables Iceberg dans le AWS Glue Data Catalog.
- Ajout de partitions – Vous n'avez pas besoin d'ajouter de partitions manuellement pour vos tables Apache Iceberg. Les nouvelles partitions au niveau des tables Apache Iceberg sont automatiquement détectées par Amazon Redshift et aucune opération manuelle n'est nécessaire pour mettre à jour les partitions dans la définition de table. Les modifications éventuellement apportées dans la spécification des partitions sont aussi appliquées automatiquement à vos requêtes sans aucune intervention de l'utilisateur.
- Ingestion des données Iceberg dans Amazon Redshift – Vous pouvez utiliser les commandes INSERT INTO ou CREATE TABLE AS pour importer les données de votre table Iceberg dans une table Amazon Redshift locale. Pour l'heure, vous ne pouvez pas utiliser la commande COPY pour ingérer le contenu d'une table Apache Iceberg dans une table Amazon Redshift locale.
- Vues matérialisées : vous pouvez créer des vues matérialisées pour les tables Apache Iceberg comme pour n'importe quelle autre table externe dans Amazon Redshift. Les éléments à prendre en considération pour les autres formats de table de lac de données valent également pour les tables Apache Iceberg. Les actualisations automatiques, la réécriture automatique des requêtes et les tables automatiques MVs sur les lacs de données ne sont actuellement pas prises en charge.
- AWS Lake Formation contrôle d'accès détaillé : Amazon Redshift AWS Lake Formation prend en charge le contrôle d'accès détaillé sur les tables Apache Iceberg.
- Paramètres de gestion des données définis par l'utilisateur – Amazon Redshift prend en charge les paramètres de gestion des données définis par l'utilisateur pour les tables Apache Iceberg. L'utilisation de paramètres de gestion des données définis par l'utilisateur sur des fichiers existants permet de personnaliser les données interrogées dans les tables externes afin d'éviter des erreurs d'analyse. Ces paramètres offrent la possibilité de gérer les incohérences entre le schéma de table et les données réelles des fichiers. Vous pouvez également utiliser les paramètres de gestion des données définis par l'utilisateur pour les tables Apache Iceberg.
- Requêtes Time Travel – Les requêtes Time Travel ne sont actuellement pas prises en charge avec les tables Apache Iceberg.
- Tarification : lorsque vous accédez à des tables Iceberg depuis un cluster, vous êtes facturé au tarif Redshift Spectrum. Lorsque vous accédez à des tables Iceberg depuis un groupe de travail, vous

êtes facturé au tarif Redshift sans serveur. Pour en savoir plus sur la tarification Redshift Spectrum et Redshift sans serveur, consultez Tarification Amazon Redshift.

Types de données pris en charge avec les tables Apache Iceberg

Cette rubrique décrit les types de données pris en charge que Redshift Spectrum peut lire à partir de tables au format Apache Iceberg.

Amazon Redshift peut interroger les tables Iceberg qui contiennent les types de données suivants :

```
binary
boolean
date
decimal
double
float
int
list
long
map
string
struct
timestamp without time zone
```

Pour en savoir plus sur les types de données Iceberg, consultez Schemas for Iceberg dans la documentation Apache Iceberg.

Le tableau suivant montre la relation qui existe entre les types de données Amazon Redshift et les types de données des tables Iceberg.

Type Iceberg	Type Amazon Redshift	Remarques
`boolean`	`boolean`	
-	`tinyint`	Non pris en charge pour les tables Iceberg dans Amazon Redshift.
-	`smallint`	Non pris en charge pour les tables Iceberg dans Amazon Redshift.

Type Iceberg	Type Amazon Redshift	Remarques
`int`	`int`	Dans les instructions SQL d'Amazon Redshift, ce type est `INTEGER`.
`long`	`bigint`	
`double`	`double precision`	
`float`	`real`	
`decimal(P, S)`	`decimal(P, S)`	`P` est la précision, `S` est l'échelle.
-	`char`	Non pris en charge pour les tables Iceberg dans Redshift Spectrum.
`string`	`string`	Dans les instructions SQL d'Amazon Redshift, ce type est `VARCHAR`.
`binary`	`binary`	
`date`	`date`	
`time`	-	
`timestamp`	`timestamp`	
`timestamptz`	-	
`list<E>`	`array`	
`map<K,V>`	`map`	

Type Iceberg	Type Amazon Redshift	Remarques
`struct<...>`	`struct`	
`fixed(L)`	-	Le type `fixed(L)` n'est actuellement pas pris en charge dans Redshift Spectrum.

Pour en savoir plus sur les types de données Amazon Redshift, consultez Types de données.

Performances des requêtes Amazon Redshift Spectrum

Cette rubrique explique comment améliorer les performances des requêtes Redshift Spectrum.

Observez le plan de requête afin de déterminer quelles étapes ont été transmises à la couche Amazon Redshift Spectrum.

Les étapes ci-après sont liées à la requête Redshift Spectrum :

- S3 Seq Scan
- S3 HashAggregate
- S3 Query Scan
- Scan Seq PartitionInfo
- Partition Loop

L'exemple suivant présente le plan d'une requête qui joint une table externe à une table locale. Notez les HashAggregate étapes S3 Seq Scan et S3 qui ont été exécutées sur les données d'Amazon S3.

```
explain
select top 10 spectrum.sales.eventid, sum(spectrum.sales.pricepaid)
from spectrum.sales, event
where spectrum.sales.eventid = event.eventid
and spectrum.sales.pricepaid > 30
group by spectrum.sales.eventid
```

```
order by 2 desc;
```

```
QUERY PLAN

------------------------------------------------------------------------------
XN Limit  (cost=1001055770628.63..1001055770628.65 rows=10 width=31)

  ->  XN Merge  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)

        Merge Key: sum(sales.derived_col2)

        ->  XN Network  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)

              Send to leader

              ->  XN Sort  (cost=1001055770628.63..1001055770629.13 rows=200 width=31)

                    Sort Key: sum(sales.derived_col2)

                    ->  XN HashAggregate  (cost=1055770620.49..1055770620.99 rows=200
 width=31)

                          ->  XN Hash Join DS_BCAST_INNER  (cost=3119.97..1055769620.49
 rows=200000 width=31)

                                Hash Cond: ("outer".derived_col1 = "inner".eventid)

                                ->  XN S3 Query Scan sales  (cost=3010.00..5010.50
 rows=200000 width=31)

                                      ->  S3 HashAggregate  (cost=3010.00..3010.50
 rows=200000 width=16)
```

```
                                          ->  S3 Seq Scan spectrum.sales
location:"s3://redshift-downloads/tickit/spectrum/sales" format:TEXT
(cost=0.00..2150.00 rows=172000 width=16)
                                                Filter: (pricepaid > 30.00)


                               ->  XN Hash  (cost=87.98..87.98 rows=8798 width=4)


                                     ->  XN Seq Scan on event  (cost=0.00..87.98
rows=8798 width=4)
```

Notez les éléments suivants dans le plan de requête :

- Le nœud `S3 Seq Scan` présente le filtre `pricepaid > 30.00` qui a été traité dans la couche Redshift Spectrum.

  Un nœud de filtre sous le nœud `XN S3 Query Scan` indique le traitement des prédicats dans Amazon Redshift au-dessus des données renvoyées depuis la couche Redshift Spectrum.
- Le nœud `S3 HashAggregate` référence le groupement dans la couche Redshift Spectrum correspondant au groupe par la clause (`group by spectrum.sales.eventid`).

Pour améliorer les performances de Redshift Spectrum, procédez comme suit :

- Utilisez des fichiers de données au format Apache Parquet. Parquet stocke les données sous forme de colonnes, de sorte que Redshift Spectrum peut éliminer les colonnes inutiles de l'analyse. Si les données sont au format texte, Redshift Spectrum doit analyser l'intégralité du fichier.
- Utilisez plusieurs fichiers afin d'optimiser le traitement parallèle, Utilisez des fichiers d'une taille supérieure à 64 Mo. Faites en sorte que tous les fichiers aient à peu près la même taille afin que la charge de travail soit équitablement distribuée. Pour plus d'informations sur les fichiers Apache Parquet et les recommandations de configuration, voir Format de fichier : configurations dans la documentation d'Apache Parquet.
- Utilisez dans vos requêtes aussi peu de colonnes que possible.
- Placez vos grandes tables de faits dans Amazon S3 et conservez vos tables de dimensions plus petites et fréquemment utilisées dans votre base de données Amazon Redshift locale.
- Mettez à jour les statistiques de table externe en définissant le paramètre numRows de TABLE PROPERTIES. Utilisez CREATE EXTERNAL TABLE ou ALTER TABLE pour définir le paramètre TABLE PROPERTIES numRows afin de refléter le nombre de lignes du tableau. Amazon Redshift

n'analyse pas des tables externes pour générer les statistiques de table utilisées par l'optimiseur de requête afin de générer un plan de requête. Si des statistiques de table ne sont pas définies pour une table externe, Amazon Redshift génère un plan d'exécution de requête. Amazon Redshift génère ce plan en partant du principe que les tables externes sont les plus grandes tables et les tables locales les plus petites.

- Le planificateur de requêtes Amazon Redshift pousse les prédicats et les agrégations vers la couche de requêtes Redshift Spectrum lorsque cela est possible. Lorsque de grandes quantités de données sont renvoyées depuis Amazon S3, le traitement est limité par les ressources de votre cluster. Redshift Spectrum procède à un dimensionnement automatique afin de traiter les requêtes volumineuses. Ainsi, les performances globales s'améliorent chaque fois que vous pouvez transmettre le traitement à la couche Redshift Spectrum.
- Ecrivez les requêtes de façon à utiliser des filtres et des regroupements qui peuvent être transmis à la couche Redshift Spectrum.

  Voici quelques exemples d'opérations pouvant être transmises à la couche Redshift Spectrum :

  - Clauses GROUP BY
  - Conditions de comparaison ou de correspondance de modèle, telles que LIKE.
  - Fonctions d'agrégation, telles que COUNT, SUM, AVG, MIN et MAX.
  - Fonctions de chaîne.

  Les opérations DISTINCT et ORDER BY, notamment, ne peuvent pas être transmises à la couche Redshift Spectrum.
- Utilisez des partitions afin de limiter les données analysées. Partitionnez les données en fonction des prédicats de requête les plus courants, puis réduisez les partitions en filtrant leurs colonnes. Pour de plus amples informations, consultez Partitionnement des tables externes Redshift Spectrum.

  Interrogez SVL_S3PARTITION afin d'afficher le total des partitions et les partitions qualifiées.
- Utilisez AWS Glue le générateur de statistiques pour calculer les statistiques au niveau des colonnes pour AWS Glue Data Catalog les tables. Une fois les statistiques AWS Glue générées pour les tables du catalogue de données, Amazon Redshift Spectrum utilise automatiquement ces statistiques pour optimiser le plan de requête. Pour plus d'informations sur le calcul des statistiques au niveau des colonnes à l'aide de cette AWS Glue méthode, consultez la section Utilisation des statistiques des colonnes dans le Guide du AWS Glue développeur.

Options de traitement des données

Cette rubrique décrit comment configurer la façon dont Redshift Spectrum gère les données dans des formats inattendus.

Vous pouvez définir des paramètres de table lorsque vous créez des tables externes pour adapter les données interrogées dans des tables externes. Sinon, des erreurs d'analyse peuvent survenir. Pour plus d'informations sur ces propriétés, consultez TABLE PROPERTIES dans CREATE EXTERNAL TABLE. Pour obtenir des exemples, consultez Exemples de gestion des données. Pour obtenir la liste des erreurs, consultez SVL_SPECTRUM_SCAN_ERROR

Vous pouvez définir les TABLE PROPERTIES (propriétés de table) suivantes lorsque vous créez des tables externes afin de spécifier la gestion des entrées pour les données interrogées dans des tables externes.

- `column_count_mismatch_handling` pour indiquer si le fichier contient moins ou plus de valeurs pour une ligne que le nombre de colonnes spécifié dans la définition de la table externe.
- `invalid_char_handling` pour spécifier la gestion des entrées pour les caractères non valides dans les colonnes contenant des données VARCHAR, CHAR et chaîne. Lorsque vous spécifiez REPLACE pour `invalid_char_handling`, vous pouvez spécifier le caractère de remplacement à utiliser.
- `numeric_overflow_handling` pour spécifier la gestion du dépassement de distribution dans les colonnes contenant des données entières et décimales.
- `surplus_bytes_handling` pour spécifier la gestion des entrées pour les octets excédentaires dans les colonnes contenant des données VARBYTE.
- `surplus_char_handling` pour spécifier la gestion des entrées pour les caractères excédentaires dans les colonnes contenant des données VARCHAR, CHAR et chaîne.

Vous pouvez définir une option de configuration pour annuler les requêtes qui dépassent un nombre maximal d'erreurs. Pour de plus amples informations, veuillez consulter spectrum_query_maxerror.

Exemple : exécution de sous-requêtes corrélées dans Redshift Spectrum

Cette rubrique décrit comment effectuer des sous-requêtes corrélées dans Redshift Spectrum. Une sous-requête corrélée est une requête qui utilise les valeurs de la requête externe.

Vous pouvez exécuter des sous-requêtes corrélées dans Redshift Spectrum. La pseudo-colonne `$spectrum_oid` permet d'effectuer des requêtes corrélées avec Redshift Spectrum. Pour exécuter une sous-requête corrélée, la pseudo-colonne `$spectrum_oid` doit être activée mais n'apparaît pas dans l'instruction SQL. Pour de plus amples informations, veuillez consulter Pseudocolonnes.

Pour créer le schéma externe et les tables externes pour cet exemple, consultez Mise en route avec Amazon Redshift Spectrum.

Vous trouverez ci-après un exemple de sous-requête corrélée dans Redshift Spectrum.

```
select *
from myspectrum_schema.sales s
where exists
( select *
from myspectrum_schema.listing l
where l.listid = s.listid )
order by salesid
limit 5;
```

```
salesid  listid   sellerid   buyerid   eventid   dateid  qtysold  pricepaid
 commission   saletime
1        1        36861      21191     7872      1875    4        728         109.2
    2008-02-18 02:36:48
2        4        8117       11498     4337      1983    2        76          11.4
    2008-06-06 05:00:16
3        5        1616       17433     8647      1983    2        350         52.5
    2008-06-06 08:26:17
4        5        1616       19715     8647      1986    1        175         26.25
    2008-06-09 08:38:52
5        6        47402      14115     8240      2069    2        154         23.1
    2008-08-31 09:17:02
```

Mesures dans Amazon Redshift Spectrum

Cette rubrique décrit les vues système que vous pouvez utiliser pour surveiller les requêtes Redshift Spectrum.

Vous pouvez surveiller les requêtes Amazon Redshift Spectrum à l'aide des vues système suivantes :

- SVL_S3QUERY

Utilisez la vue SVL_S3QUERY afin d'obtenir des détails sur les requêtes Redshift Spectrum (requêtes Amazon S3) au niveau du segment et de la tranche de nœud.

- SVL_S3QUERY_SUMMARY

  Utilisez la vue SVL_S3QUERY_SUMMARY pour obtenir un résumé de toutes les requêtes Amazon Redshift Spectrum (requêtes S3) qui ont été exécutées sur le système.

Voici quelques-uns des éléments à rechercher dans SVL_S3QUERY_SUMMARY :

- Nombre de fichiers traités par la requête Redshift Spectrum.
- Le nombre d'octets analysés à partir d'Amazon S3. Le coût d'une requête Redshift Spectrum est répercuté dans la quantité de données analysées depuis Amazon S3.
- Nombre d'octets retournés par la couche Redshift Spectrum au cluster. Si le volume de données renvoyées est important, les performances du système peuvent être affectées.
- Durées maximale et moyenne des requêtes Redshift Spectrum. Les requêtes de longue durée peuvent être dues à un goulot d'étranglement.

Résolution des problèmes liés aux requêtes dans Amazon Redshift Spectrum

Cette rubrique est une référence pour les problèmes courants que vous pouvez rencontrer avec les requêtes Amazon Redshift Spectrum.

Pour afficher les erreurs générées par les requêtes Redshift Spectrum, interrogez la table système SVL_S3LOG.

Rubriques

- Dépassement des nouvelles tentatives
- Accès limité
- Limite de ressource dépassée
- Aucune ligne retournée pour une table partitionnée
- Erreur Non autorisé
- Formats de données incompatibles

- Erreur de syntaxe lors de l'utilisation de Hive DDL dans Amazon Redshift
- Autorisation de créer des tables temporaires
- Plage non valide
- Numéro de version Parquet non valide

Dépassement des nouvelles tentatives

En cas d'expiration d'une requête Amazon Redshift Spectrum, celle-ci est annulée puis soumise à nouveau. Après cinq tentatives, la requête échoue avec l'erreur suivante.

```
error:  Spectrum Scan Error: Retries exceeded
```

Les causes possibles sont notamment les suivantes :

- Fichiers de taille volumineuse (supérieure à 1 Go). Vérifiez la taille de vos fichiers dans Amazon S3 et recherchez les fichiers volumineux et la distorsion de la taille des fichiers. Fractionnez les fichiers volumineux en fichiers plus petits, d'une taille comprise entre 100 Mo et 1 Go. Faites en sorte que les fichiers aient à peu près la même taille.
- Débit lent du réseau. Réessayez la requête plus tard.

Accès limité

Amazon Redshift Spectrum est soumis aux quotas de service des AWS autres services. En cas d'utilisation élevée, les requêtes Redshift Spectrum peuvent être amenées à ralentir, ce qui entraîne l'erreur suivante.

```
error:  Spectrum Scan Error: Access throttled
```

Deux types de limitation peuvent se produire :

- Accès limité par Amazon S3.
- Accès limité par. AWS KMS

Le contexte d'erreur fournit plus de détails sur le type de limitation. Vous trouverez ci-dessous les causes et les résolutions possibles pour cette limitation.

Accès limité par Amazon S3

Amazon S3 peut bloquer une demande de Redshift Spectrum si le taux de demande de lecture sur un préfixe est trop élevé. Pour plus d'informations sur le taux de GET/HEAD demandes que vous pouvez atteindre dans Amazon S3, consultez le guide de l'utilisateur d'Amazon Simple Storage Service sur l'optimisation des performances d'Amazon S3. Le taux de GET/HEAD demandes Amazon S3 prend en compte toutes les GET/HEAD demandes sur un préfixe, de sorte que les différentes applications accédant au même préfixe partagent le taux total de demandes.

Si vos demandes Redshift Spectrum sont fréquemment limitées par Amazon S3, réduisez le nombre de demandes Amazon S3 GET/HEAD que Redshift Spectrum envoie à Amazon S3. Pour ce faire, essayez de fusionner de petits fichiers dans des fichiers plus volumineux. Nous vous recommandons d'utiliser des fichiers de 64 Mo ou plus.

Envisagez également de partitionner vos tables Redshift Spectrum pour bénéficier d'un filtrage précoce et réduire le nombre de fichiers accédés dans Amazon S3. Pour de plus amples informations, consultez Partitionnement des tables externes Redshift Spectrum.

Accès limité par AWS KMS

Si vous stockez vos données dans Amazon S3 à l'aide du chiffrement côté serveur (SSE-S3 ou SSE-KMS), Amazon S3 appelle une opération d'API AWS KMS pour chaque fichier auquel Redshift Spectrum accède. Ces demandes sont prises en compte dans votre quota d'opérations cryptographiques ; pour de plus amples informations, consultez Quotas de demande AWS KMS. Pour plus d'informations sur SSE-S3 et SSE-KMS, voir Protection des données à l'aide du chiffrement côté serveur et Protection des données à l'aide du chiffrement côté serveur avec des clés KMS stockées dans AWS KMS dans le Guide de l'utilisateur Amazon Simple Storage Service.

La première étape pour réduire le nombre de demandes adressées par Redshift Spectrum AWS KMS consiste à réduire le nombre de fichiers consultés. Pour ce faire, essayez de fusionner de petits fichiers dans des fichiers plus volumineux. Nous vous recommandons d'utiliser des fichiers de 64 Mo ou plus.

Si vos demandes Redshift Spectrum sont fréquemment limitées AWS KMS, envisagez de demander une augmentation du quota pour votre taux de demandes pour les opérations AWS KMS cryptographiques. Pour demander une augmentation de quota, consultez les Limites de service AWS dans le Référence générale d'Amazon Web Services.

Limite de ressource dépassée

Redshift Spectrum applique une limite supérieure à la quantité de mémoire qu'une requête peut utiliser. Une requête Redshift Spectrum qui nécessite plus de mémoire échoue, ce qui entraîne l'erreur suivante.

```
error:  Spectrum Scan Error: Resource limit exceeded
```

Il y a deux raisons courantes pour lesquelles une requête Redshift Spectrum peut dépasser sa capacité de mémoire :

- Redshift Spectrum traite une grande partie de données qui ne peut pas être divisée en sections plus petites.
- Une étape d'agrégation importante est traitée par Redshift Spectrum.

Nous vous recommandons d'utiliser un format de fichier prenant en charge les lectures parallèles avec des tailles fractionnées de 128 Mo ou moins. Consultez Fichiers de données pour les requêtes dans Amazon Redshift Spectrum pour connaître les formats de fichiers pris en charge et les instructions génériques pour la création de fichiers de données. Lorsque vous utilisez des formats de fichier ou des algorithmes de compression qui ne prennent pas en charge les lectures parallèles, nous vous recommandons de conserver les tailles de fichier entre 64 et 128 Mo.

Aucune ligne retournée pour une table partitionnée

Si votre requête ne retourne aucune ligne depuis une table externe partitionnée, vérifiez si une partition a été ajoutée pour cette table externe. Redshift Spectrum analyse uniquement les fichiers dans un emplacement Amazon S3 qui a été explicitement ajouté à l'aide de l'instruction `ALTER TABLE … ADD PARTITION`. Pour afficher les partitions existantes, interrogez la vue SVV_EXTERNAL_PARTITIONS. Exécutez `ALTER TABLE … ADD PARTITION` pour chaque partition manquante.

Erreur Non autorisé

Vérifiez que le rôle IAM pour le cluster autorise l'accès aux objets fichiers Amazon S3. Si votre base de données externe se trouve sur Amazon Athena, vérifiez que le rôle IAM autorise l'accès aux ressources Athena. Pour de plus amples informations, consultez Politiques IAM pour Amazon Redshift Spectrum.

Formats de données incompatibles

Dans les fichiers en colonnes (fichiers Apache Parquet par exemple), le type de colonne est incorporé aux données. Dans la définition de CREATE EXTERNAL TABLE, le type de colonne doit correspondre au type de colonne du fichier de données. En cas de décalage, vous recevez un message d'erreur similaire au suivant :

```
File 'https://s3bucket/location/file has an incompatible Parquet schema
for column ‘s3://s3bucket/location.col1'. Column type: VARCHAR, Par
```

Le message d'erreur peut être tronqué en raison de la limite de la longueur du message. Pour récupérer le message d'erreur complet, notamment le nom et le type de colonne, interrogez la vue système SVL_S3LOG.

L'exemple suivant interroge SVL_S3LOG par rapport à la dernière requête exécutée.

```
select message
from svl_s3log
where query = pg_last_query_id()
order by query,segment,slice;
```

Voici un exemple de résultat affichant le message d'erreur complet.

```
                            message
-----------------------------------------------------------------------------------------
Spectrum Scan Error. File 'https://s3bucket/location/file has an incompatible
Parquet schema for column ' s3bucket/location.col1'.
Column type: VARCHAR, Parquet schema:\noptional int64 l_orderkey [i:0 d:1 r:0]\n
```

Pour corriger l'erreur, modifiez la table externe afin qu'elle corresponde au type de colonne du fichier Parquet.

Erreur de syntaxe lors de l'utilisation de Hive DDL dans Amazon Redshift

Amazon Redshift prend en charge le langage de définition de données (DDL) pour CREATE EXTERNAL TABLE qui est similaire au DDL de Hive. Cependant, les deux types de DDL ne sont pas toujours exactement identiques. Ainsi, des erreurs de syntaxe peuvent se produire si vous copiez le langage Hive DDL en vue de créer ou de modifier des tables externes Amazon Redshift. Voici quelques exemples de différences entre Amazon Redshift et Hive DDL :

- Amazon Redshift exige des apostrophes droites (’) alors que Hive DDL prend en charge les guillemets droits (").
- Amazon Redshift ne prend pas en charge le type de données STRING. Utilisez plutôt le type VARCHAR.

Autorisation de créer des tables temporaires

Pour exécuter des requêtes Redshift Spectrum, l’utilisateur de la base de données doit avoir l’autorisation d’y créer des tables temporaires. L’exemple suivant accorde une autorisation temporaire concernant la base de données `spectrumdb` au groupe d’utilisateurs `spectrumusers`.

```
grant temp on database spectrumdb  to group spectrumusers;
```

Pour de plus amples informations, veuillez consulter GRANT.

Plage non valide

Redshift Spectrum s’attend à ce que les fichiers dans Amazon S3 appartenant à une table externe ne soient pas écrasés pendant une requête. Dans cela se produit, cela peut entraîner l’erreur suivante.

```
Error: HTTP response error code: 416 Message: InvalidRange The requested range is not
 satisfiable
```

Pour éviter cette erreur, vérifiez que les fichiers Amazon S3 ne sont pas écrasés lorsqu’ils sont interrogés avec Redshift Spectrum.

Numéro de version Parquet non valide

Redshift Spectrum vérifie les métadonnées de chaque fichier Apache Parquet auquel il accède. Si la vérification échoue, cela peut entraîner une erreur semblable à celle-ci :

```
File 'https://s3.region.amazonaws.com/s3bucket/location/file has an invalid version
 number
```

Deux raisons courantes peuvent provoquer l’échec de la vérification :

- Le fichier Parquet a été écrasé pendant la requête (consultez Plage non valide).

- Le fichier Parquet est corrompu.

Didacticiel : Faire une requête de données imbriquées avec Amazon Redshift Spectrum

Ce didacticiel explique comment interroger des données imbriquées avec Redshift Spectrum. Les données imbriquées sont des données qui contiennent des champs imbriqués. Les champs imbriqués sont des champs assemblés en une seule entité, tels que des tableaux, des structures ou des objets.

Rubriques

- Présentation
- Étape 1 : Création d'un tableau externe contenant des données imbriquées
- Étape 2 : Faire une requête de vos données imbriquées dans Amazon S3 avec des extensions SQL
- Cas d'utilisation de données imbriquées
- Limitations des données imbriquées (version préliminaire)
- Sérialisation de JSON imbriqué complexe

Présentation

Amazon Redshift Spectrum prend en charge la requête de données imbriquées dans Parquet, ORC, JSON et les formats de fichier Ion. Redshift Spectrum accède aux données à l'aide de tableaux externes. Vous pouvez créer des tableaux externes qui utilisent les types de données complexes `struct`, `array`, et `map`.

Par exemple, supposons que votre fichier de données contienne les données suivantes dans Amazon S3 dans un fichier nommé `customers`. Bien qu'il n'y ait aucun élément racine, chaque objet JSON de cet exemple de données représente une ligne d'une table.

```
{"id": 1,
 "name": {"given": "John", "family": "Smith"},
 "phones": ["123-457789"],
 "orders": [{"shipdate": "2018-03-01T11:59:59.000Z", "price": 100.50},
            {"shipdate": "2018-03-01T09:10:00.000Z", "price": 99.12}]
```

```
}
{"id": 2,
 "name": {"given": "Jenny", "family": "Doe"},
 "phones": ["858-8675309", "415-9876543"],
 "orders": []
}
{"id": 3,
 "name": {"given": "Andy", "family": "Jones"},
 "phones": [],
 "orders": [{"shipdate": "2018-03-02T08:02:15.000Z", "price": 13.50}]
}
```

Vous pouvez utiliser Amazon Redshift Spectrum pour soumettre une requête sur les données imbriquées dans des fichiers. Le didacticiel suivant vous montre comment effectuer cette opération avec les données Apache Parquet.

Prérequis

Si vous n'utilisez pas encore Redshift Spectrum, suivez les étapes détaillées dans Mise en route avec Amazon Redshift Spectrum avant de poursuivre.

Pour créer un schéma externe, remplacez l'ARN de rôle IAM dans la commande suivante par l'ARN de rôle que vous avez créé dans Créer un rôle IAM. Ensuite, exécutez la commande dans votre client SQL.

```
create external schema spectrum
from data catalog
database 'myspectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myspectrum_role'
create external database if not exists;
```

Étape 1 : Création d'un tableau externe contenant des données imbriquées

Vous pouvez afficher les données source en les téléchargeant depuis Amazon S3.

Exécutez la commande suivante pour créer un tableau externe pour ce didacticiel.

```
CREATE EXTERNAL TABLE spectrum.customers (
  id     int,
  name   struct<given:varchar(20), family:varchar(20)>,
  phones array<varchar(20)>,
```

```
  orders array<struct<shipdate:timestamp, price:double precision>>
)
STORED AS PARQUET
LOCATION 's3://redshift-downloads/tickit/spectrum/customers/';
```

Dans l'exemple précédent, le tableau externe `spectrum.customers` utilise les types de données `struct` et `array` pour définir les colonnes dotées de données imbriquées. Amazon Redshift Spectrum prend en charge la requête de données imbriquées dans Parquet, ORC, JSON et les formats de fichier Ion. Le paramètre `STORED AS` est `PARQUET` pour les fichiers Apache Parquet. Le paramètre `LOCATION` doit se référer au dossier Amazon S3 contenant les données ou fichiers imbriqués. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL TABLE.

Vous pouvez imbriquer les types `array` et `struct` à n'importe quel niveau. Vous pouvez par exemple définir une colonne nommée `toparray`, comme l'illustre l'exemple suivant.

```
toparray array<struct<nestedarray:
         array<struct<morenestedarray:
         array<string>>>>>
```

Vous pouvez également imbriquer les types `struct` comme l'illustre la colonne `x` dans l'exemple suivant.

```
x struct<a: string,
         b: struct<c: integer,
                   d: struct<e: string>
                  >
        >
```

Étape 2 : Faire une requête de vos données imbriquées dans Amazon S3 avec des extensions SQL

Redshift Spectrum prend en charge les requêtes de types complexes `array`, `map` et `struct` via des extensions à la syntaxe SQL Amazon Redshift.

Extension 1 : Accès aux colonnes de structs

Vous pouvez extraire des données à partir des colonnes `struct` à l'aide d'une notation par points qui connecte les noms de champs en chemins. Par exemple, la requête suivante retourne les noms

et prénoms de clients. Le prénom est obtenu via le long chemin `c.name.given`. Le nom de famille est obtenu via le long chemin `c.name.family`.

```
SELECT c.id, c.name.given, c.name.family
FROM   spectrum.customers c;
```

La requête précédente renvoie les données suivantes.

```
id | given | family
---|-------|-------
1  | John  | Smith
2  | Jenny | Doe
3  | Andy  | Jones
(3 rows)
```

Un `struct` peut être une colonne d'un autre `struct`, qui peut être une colonne d'un autre `struct` et ce à tout niveau. Les chemins qui accèdent aux colonnes dans des `struct` aussi profondément imbriqués peuvent être très longs. Par exemple, regardez la définition pour la colonne `x` dans l'illustration suivante.

```
x struct<a: string,
         b: struct<c: integer,
                   d: struct<e: string>
                  >
        >
```

Vous pouvez accéder aux données dans `e` en tant que `x.b.d.e`.

Extension 2 : Surplomber les pans d'une clause FROM

Vous pouvez extraire des données de colonnes `array` (et, par extension, des colonnes `map`) en précisant les colonnes `array` dans une clause `FROM` à la place des noms de tableaux. L'extension s'applique à la clause `FROM` de la requête principale, ainsi qu'aux clauses `FROM` des sous-requêtes.

Vous pouvez référencer les éléments `array` d'après leur position, comme `c.orders[0]` (version préliminaire).

En combinant le surplombage de `arrays` avec des raccords, vous pouvez réaliser différentes sortes de désimbrication, comme les cas d'utilisation suivants l'expliquent.

Désimbrication à l'aide de raccords internes

La requête suivante sélectionne les dates d'expédition IDs des clients et des commandes pour les clients qui ont des commandes. L'extension SQL dans la clause FROM `c.orders o` dépend de l'alias `c`.

```
SELECT c.id, o.shipdate
FROM   spectrum.customers c, c.orders o
```

Pour chaque `c` de client disposant de commandes, la clause `FROM` renvoie une ligne pour chaque commande `o` du client `c`. Cette ligne combine la ligne du client `c` et la ligne de commande `o`. Ensuite, la clause `SELECT` garde uniquement le `c.id` et le `o.shipdate`. Le résultat est le suivant.

```
id|      shipdate
--|----------------------
1 |2018-03-01  11:59:59
1 |2018-03-01  09:10:00
3 |2018-03-02  08:02:15
(3 rows)
```

L'alias `c` fournit un accès aux champs du client et l'alias `o` fournit un accès aux champs de commande.

La sémantique est similaire au SQL standard. Vous pouvez envisager la clause `FROM` comme exécutant la boucle imbriquée suivante qui est suivie par la sélection de champs pour la sortie par `SELECT`.

```
for each customer c in spectrum.customers
  for each order o in c.orders
     output c.id and o.shipdate
```

Ainsi, si un client ne dispose pas de commande, le client n'apparaît pas dans le résultat.

Vous pouvez également envisager ceci comme une clause `FROM` qui exécute un `JOIN` avec le tableau `customers` et le pan `orders`. Dans les faits, vous pouvez également rédiger la requête comme l'illustre l'exemple suivant.

```
SELECT c.id, o.shipdate
FROM   spectrum.customers c INNER JOIN c.orders o ON true
```

> **Note**
>
> Si un schéma nommé `c` existe avec un tableau nommé `orders`, alors `c.orders` se réfère au tableau `orders`, et non à la colonne de pan `customers`.

Désimbrication à l'aide de raccords gauches

La requête suivante sort tous les noms des clients et leurs commandes. Si un client n'a pas encore placé de commande, son nom sera tout de même renvoyé. Cependant, dans ce cas, les colonnes de commande sont NULLES, comme l'illustre l'exemple suivant pour Jenny Doe.

```
SELECT c.id, c.name.given, c.name.family, o.shipdate, o.price
FROM   spectrum.customers c LEFT JOIN c.orders o ON true
```

La requête précédente renvoie les données suivantes.

```
id  |  given  | family  |    shipdate          | price
----|---------|---------|----------------------|--------
 1  |  John   | Smith   | 2018-03-01  11:59:59 | 100.5
 1  |  John   | Smith   | 2018-03-01  09:10:00 |  99.12
 2  |  Jenny  | Doe     |                      |
 3  |  Andy   | Jones   | 2018-03-02  08:02:15 |  13.5
 (4 rows)
```

Extension 3 : Obtenir directement à un pan de scalaires en utilisant un alias

Lorsqu'un alias p dans une clause FROM surplombe tout un tableau de scalaires, la requête se réfère aux valeurs de p en tant que p. Par exemple, la requête suivante produit des paires de noms de clients et de numéros de téléphone.

```
SELECT c.name.given, c.name.family, p AS phone
FROM   spectrum.customers c LEFT JOIN c.phones p ON true
```

La requête précédente renvoie les données suivantes.

```
given  |  family  |  phone
-------|----------|-----------
John   | Smith    | 123-4577891
Jenny  | Doe      | 858-8675309
```

```
Jenny  | Doe      | 415-9876543
Andy   | Jones    |
(4 rows)
```

Extension 4 : Accès aux éléments de cartes

Redshift Spectrum traite le type de donnée `map` comme un type de `array` contenant des types `struct` avec une colonne `key` et une colonne `value`. La `key` doit être `scalar`; la valeur peut être de n'importe quel type de donnée.

Par exemple, le code suivant crée un tableau externe avec une `map` pour stocker les numéros de téléphone.

```
CREATE EXTERNAL TABLE spectrum.customers2 (
  id     int,
  name   struct<given:varchar(20), family:varchar(20)>,
  phones map<varchar(20), varchar(20)>,
  orders array<struct<shipdate:timestamp, price:double precision>>
)
STORED AS PARQUET
LOCATION 's3://redshift-downloads/tickit/spectrum/customers/';
```

Parce qu'un type de `map` se comporte comme un type de `array` avec des colonnes `key` et `value`, vous pouvez penser aux schémas précédents comme s'ils étaient les suivants.

```
CREATE EXTERNAL TABLE spectrum.customers3 (
  id     int,
  name   struct<given:varchar(20), family:varchar(20)>,
  phones array<struct<key:varchar(20), value:varchar(20)>>,
  orders array<struct<shipdate:timestamp, price:double precision>>
)
STORED AS PARQUET
LOCATION 's3://redshift-downloads/tickit/spectrum/customers/';
```

La requête suivante renvoie les noms de clients avec un numéro de téléphone mobile et renvoie le numéro pour chaque nom. La requête de carte est traitée comme l'équivalent d'une requête de `array` imbriquée de types `struct`. La requête suivante ne renvoie des données que si vous avez créé le tableau externe précédemment évoqué.

```
SELECT c.name.given, c.name.family, p.value
```

```
FROM   spectrum.customers c, c.phones p
WHERE  p.key = 'mobile';
```

> **Note**
>
> La `key` pour une `map` est une `string` pour les types de fichiers Ion et JSON.

Cas d'utilisation de données imbriquées

Cette rubrique décrit les cas d'utilisation des données imbriquées. Les données imbriquées sont des données qui contiennent des champs imbriqués. Les champs imbriqués sont des champs assemblés en une seule entité, tels que des tableaux, des structures ou des objets.

Vous pouvez combiner les extensions précédemment décrites avec les caractéristiques SQL habituelles. Les cas d'utilisation suivants illustrent certaines combinaisons courantes. Ces exemples aident à illustrer comment utiliser des données imbriquées. Elles ne font pas partie du didacticiel.

Rubriques

- Ingérer des données imbriquées
- Agréger des données imbriquées avec des sous-requêtes
- Joindre Amazon Redshift et des données imbriquées

Ingérer des données imbriquées

Vous pouvez utiliser une déclaration `CREATE TABLE AS` pour ingérer des données à partir d'un tableau externe contenant des types de données complexes. La requête suivante extrait tous les clients et leurs numéros de téléphone à partir du tableau externe, à l'aide de `LEFT JOIN` et les stocke dans le tableau Amazon Redshift `CustomerPhones`.

```
CREATE TABLE CustomerPhones AS
SELECT  c.name.given, c.name.family, p AS phone
FROM    spectrum.customers c LEFT JOIN c.phones p ON true;
```

Agréger des données imbriquées avec des sous-requêtes

Vous pouvez utiliser une sous-requête pour agréger des données imbriquées. L'exemple suivant illustre cette approche.

```
SELECT c.name.given, c.name.family, (SELECT COUNT(*) FROM c.orders o) AS ordercount
FROM   spectrum.customers c;
```

Les données suivantes sont renvoyées.

```
given   |  family  |  ordercount
--------|----------|-------------
 Jenny  |  Doe     |       0
 John   |  Smith   |       2
 Andy   |  Jones   |       1
 (3 rows)
```

> **Note**
>
> La manière la plus efficace d'agréger des données imbriquées en les groupant par ligne parent est illustrée dans l'exemple précédent. Dans cet exemple, les lignes imbriquées de `c.orders` sont groupées par leur ligne parent `c`. Autrement, si vous savez que l'`id` est unique pour chaque `customer` et que `o.shipdate` n'a jamais une valeur nulle, vous pouvez agréger de la manière illustrée dans l'exemple suivant. Cependant, cette approche n'est généralement pas aussi efficace que l'exemple précédent.

```
SELECT    c.name.given, c.name.family, COUNT(o.shipdate) AS ordercount
FROM      spectrum.customers c LEFT JOIN c.orders o ON true
GROUP BY  c.id, c.name.given, c.name.family;
```

Vous pouvez également rédiger la requête en utilisant une sous-requête dans la clause `FROM` qui se réfère à un alias (`c`) de la requête d'ancêtre et extrait les données de tableau. L'exemple suivant illustre cette approche.

```
SELECT c.name.given, c.name.family, s.count AS ordercount
FROM   spectrum.customers c, (SELECT count(*) AS count FROM c.orders o) s;
```

Joindre Amazon Redshift et des données imbriquées

Vous pouvez également raccorder les données Amazon Redshift aux données imbriquées dans un tableau externe. Par exemple, supposons que vous avez les données imbriquées suivantes dans Amazon S3.

```
CREATE EXTERNAL TABLE spectrum.customers2 (
  id      int,
  name    struct<given:varchar(20), family:varchar(20)>,
  phones  array<varchar(20)>,
  orders  array<struct<shipdate:timestamp, item:int>>
);
```

Supposons également que vous disposez du tableau suivant dans Amazon Redshift.

```
CREATE TABLE prices (
  id int,
  price double precision
);
```

La requête suivante trouve le nombre total et le montant de chaque achat du client selon ce qui précède. L'illustration suivante n'est qu'un exemple. Elle ne renvoie des données que si vous avez créé les tableaux précédemment évoqués.

```
SELECT   c.name.given, c.name.family, COUNT(o.date) AS ordercount, SUM(p.price) AS
 ordersum
FROM     spectrum.customers2 c, c.orders o, prices p ON o.item = p.id
GROUP BY c.id, c.name.given, c.name.family;
```

Limitations des données imbriquées (version préliminaire)

Cette rubrique décrit les limites liées à la lecture de données imbriquées avec Redshift Spectrum. Les données imbriquées sont des données qui contiennent des champs imbriqués. Les champs imbriqués sont des champs assemblés en une seule entité, tels que des tableaux, des structures ou des objets.

> **Note**
>
> Les limitations marquées (aperçu) dans la liste suivante s'appliquent uniquement aux clusters d'aperçu créés dans les régions suivantes.
>
> - USA Est (Ohio) (us-east-2)
> - USA Est (Virginie du Nord) (us-east-1)
> - USA Ouest (Californie du Nord) (us-west-1)
> - Asie-Pacifique (Tokyo) (ap-northeast-1)

- Europe (Irlande) (eu-west-1)
- Europe (Stockholm) (eu-north-1)

Pour en savoir plus sur la configuration des clusters en version préliminaire, consultez Création d'un cluster en version préliminaire dans le Guide de gestion Amazon Redshift.

Les limites suivantes s'appliquent aux données imbriquées :

- Un type `array` ou `map` peut contenir d'autres types `array` ou `map` tant que les requêtes sur les types imbriqués `arrays` ou `maps` ne renvoient pas de valeurs `scalar` (version préliminaire).
- Amazon Redshift Spectrum prend uniquement en charge les types de données complexes sous forme de tables externes.
- Les colonnes de résultats des sous-requêtes doivent être de niveau supérieur (version préliminaire).
- Si une expression `OUTER JOIN` se réfère à un tableau imbriqué, il ne peut que se référer à ce tableau et ses pans imbriqués (et ses cartes). Si une expression `OUTER JOIN` ne se réfère pas à un tableau imbriqué, il peut se référer à tout genre de tableaux non imbriqués.
- Si une clause `FROM` d'une sous-requête se réfère à un tableau imbriqué, il ne peut se référer à quelconque autre tableau.
- Si une sous-requête dépend d'une table imbriquée qui se réfère à un parent, vous ne pouvez utiliser le parent que dans la clause `FROM`. Vous ne pouvez pas utiliser le parent dans une quelconque autre clause, comme une clause `SELECT` ou `WHERE`. Par exemple, la requête suivante ne s'exécute pas, car la clause `SELECT` de la sous-requête fait référence à la table parent `c`.

```
SELECT c.name.given
FROM   spectrum.customers c
WHERE (SELECT COUNT(c.id) FROM c.phones p WHERE p LIKE '858%') > 1;
```

La requête suivante fonctionne parce que le parent `c` est utilisé uniquement dans la clause `FROM` de la sous-requête.

```
SELECT c.name.given
FROM   spectrum.customers c
WHERE (SELECT COUNT(*) FROM c.phones p WHERE p LIKE '858%') > 1;
```

- Une sous-requête qui accède à des données imbriquées à tout autre endroit que la clause FROM doit renvoyer une seule valeur. Les seules exceptions sont les opérateurs (NOT) EXISTS dans une clause WHERE.
- (NOT) IN n'est pas pris en charge.
- La profondeur d'imbrication maximum pour tous les types imbriqués est de 100. Cette restriction s'applique à tous les formats de fichier (Parquet, ORC, Ion et JSON).
- Les sous-requêtes d'agrégation qui accèdent aux données imbriquées peuvent uniquement se reporter à arrays et maps dans leur clause FROM, et non pas à une table externe.
- L'interrogation des pseudocolonnes de données imbriquées dans une table Redshift Spectrum n'est pas prise en charge. Pour de plus amples informations, veuillez consulter Pseudocolonnes.
- Quand vous extrayez des données à partir de colonnes d'un tableau ou d'une carte en les spécifiant dans une clause FROM, vous ne pouvez sélectionner des valeurs dans ces colonnes que si les valeurs sont scalar. Par exemple, les requêtes suivantes essayent toutes deux de sélectionner (SELECT) des éléments depuis l'intérieur d'un tableau. La requête qui sélectionne fonctionne arr.a, car arr.a est une valeur scalar. La deuxième requête ne fonctionne pas, car array est un tableau extrait de s3.nested table dans la clause FROM (version préliminaire).

```
SELECT array_column FROM s3.nested_table;

array_column
-----------------
[{"a":1},{"b":2}]

SELECT arr.a FROM s3.nested_table t, t.array_column arr;

arr.a
-----
1

--This query fails to run.
SELECT array FROM s3.nested_table tab, tab.array_column array;
```

Vous ne pouvez pas utiliser dans la clause FROM un tableau ni une carte qui proviennent eux-mêmes d'un autre tableau ou d'une autre carte. Pour sélectionner des tableaux ou d'autres structures complexes imbriquées dans d'autres tableaux, pensez à utiliser des index dans l'instruction SELECT.

Sérialisation de JSON imbriqué complexe

Cette rubrique explique comment sérialiser des données imbriquées au format JSON. Les données imbriquées sont des données qui contiennent des champs imbriqués. Les champs imbriqués sont des champs assemblés en une seule entité, tels que des tableaux, des structures ou des objets.

Une alternative aux méthodes présentées dans ce didacticiel consiste à interroger les colonnes de collection imbriquées de niveau supérieur en tant que JSON sérialisé. Vous pouvez utiliser la sérialisation pour inspecter, convertir et intégrer des données imbriquées au format JSON avec Redshift Spectrum. Cette méthode est prise en charge pour les formats ORC, JSON, Ion et Parquet. Utilisez le paramètre de configuration de session `json_serialization_enable` pour configurer le comportement de sérialisation. Lorsqu'ils sont définis, les types de données JSON complexes sont sérialisés en VARCHAR (65535). Le JSON imbriqué est accessible avec les Fonctions JSON. Pour de plus amples informations, veuillez consulter json_serialization_enable.

Par exemple, sans définir `json_serialization_enable`, les requêtes suivantes qui accèdent directement aux colonnes imbriquées échouent.

```
SELECT * FROM spectrum.customers LIMIT 1;

=> ERROR:  Nested tables do not support '*' in the SELECT clause.

SELECT name FROM spectrum.customers LIMIT 1;

=> ERROR:  column "name" does not exist in customers
```

Le paramètre `json_serialization_enable` permet d'interroger directement les collections de niveau supérieur.

```
SET json_serialization_enable TO true;

SELECT * FROM spectrum.customers order by id LIMIT 1;

id | name                                  | phones         | orders
---+---------------------------------------+----------------
+-----------------------------------------------------------------------------------------------
1  | {"given": "John", "family": "Smith"} | ["123-457789"] | [{"shipdate":
 "2018-03-01T11:59:59.000Z", "price": 100.50}, {"shipdate": "2018-03-01T09:10:00.000Z",
 "price": 99.12}]

SELECT name FROM spectrum.customers order by id LIMIT 1;
```

```
name
---------
{"given": "John", "family": "Smith"}
```

Tenez compte des éléments suivants lors de la sérialisation du JSON imbriqué.

- Lorsque les colonnes de collection sont sérialisées en tant que VARCHAR(65535), leurs sous-champs imbriqués ne sont pas accessibles directement dans le cadre de la syntaxe de requête (par exemple, dans la clause de filtre). Cependant, les fonctions JSON peuvent être utilisées pour accéder au JSON imbriqué.
- Les représentations spécialisées suivantes ne sont pas prises en charge :
  - Syndicats ORC
  - Cartes ORC avec clés de type complexe
  - Datagrammes Ion
  - SEXP Ion
- Les horodatages sont renvoyés sous forme de chaînes sérialisées ISO.
- Les clés de carte primitives sont promues en chaîne (par exemple, 1 sur `"1"`).
- Les valeurs nulles de niveau supérieur sont sérialisées sous la forme. NULLs
- Si la sérialisation dépasse la taille VARCHAR maximale de 65535, la cellule est définie sur NULL.

Sérialisation de types complexes contenant des chaînes JSON

Par défaut, les valeurs de chaîne contenues dans les collections imbriquées sont sérialisées en tant que chaînes JSON échappées. L'échappement peut être problématique lorsque les chaînes sont JSON valides. Au lieu de cela, vous pouvez écrire des sous-éléments ou des champs imbriqués VARCHAR directement en tant que JSON. Activez ce comportement avec la configuration au niveau de la session `json_serialization_parse_nested_strings`. Lorsque `json_serialization_enable` et `json_serialization_parse_nested_strings` sont définis, les valeurs JSON valides sont sérialisées en ligne sans caractères d'échappement. Lorsque la valeur n'est pas un JSON valide, elle est échappée comme si la valeur de configuration `json_serialization_parse_nested_strings` n'était pas définie. Pour de plus amples informations, veuillez consulter json_serialization_parse_nested_strings.

Supposons que les données de l'exemple précédent contiennent JSON en tant que type complexe `structs` dans le champ `name` VARCHAR(20) :

```
name
---------
{"given": "{\"first\":\"John\",\"middle\":\"James\"}", "family": "Smith"}
```

Lorsque `json_serialization_parse_nested_strings` est défini, la colonne `name` est sérialisée comme suit :

```
SET json_serialization_enable TO true;
SET json_serialization_parse_nested_strings TO true;
SELECT name FROM spectrum.customers order by id LIMIT 1;

name
---------
{"given": {"first":"John","middle":"James"}, "family": "Smith"}
```

Au lieu d'être échappée comme suit :

```
SET json_serialization_enable TO true;
SELECT name FROM spectrum.customers order by id LIMIT 1;

name
---------
{"given": "{\"first\":\"John\",\"middle\":\"James\"}", "family": "Smith"}
```

HyperLogLog croquis

Cette rubrique explique comment utiliser les HyperLogLog esquisses dans Amazon Redshift. HyperLogLog est un algorithme pour le problème du nombre distinct, qui donne une approximation du nombre d'éléments distincts dans un ensemble de données. HyperLogLog les esquisses sont des tableaux de données d'unicité concernant un ensemble de données.

HyperLogLogest un algorithme utilisé pour estimer la cardinalité d'un multiset. La cardinalité fait référence au nombre de valeurs distinctes dans un multi-ensemble. Par exemple, dans l'ensemble {4,3,6,2,2,6,4,3,6,2,2,3}, la cardinalité est 4 et les valeurs représentées sont 4, 3, 6 et 2.

La précision de l' HyperLogLog algorithme (également appelée valeur m) peut affecter la précision de la cardinalité estimée. Pour estimer la cardinalité, Amazon Redshift utilise une valeur de précision par défaut de 15. Cette valeur peut aller jusqu'à 26 pour les jeux de données plus petits. Ainsi, l'erreur relative moyenne se situe entre 0,01 et 0,6 %.

Lors du calcul de la cardinalité d'un multiensemble, l' HyperLogLog algorithme génère une construction appelée esquisse HLL. Une esquisse HLL encapsule des informations sur les valeurs distinctes d'un multi-ensemble. Le type de données Amazon Redshift HLLSKETCH représente ces valeurs d'esquisse. Ce type de données peut être utilisé pour stocker des esquisses dans une table Amazon Redshift. De plus, Amazon Redshift prend en charge les opérations pouvant être appliquées aux valeurs HLLSKETCH en tant que fonctions d'agrégation et scalaires. Vous pouvez utiliser ces fonctions pour extraire la cardinalité d'un HLLSKETCH et combiner plusieurs valeurs HLLSKETCH.

Le type de données HLLSKETCH offre des avantages significatifs quant aux performances des requêtes lors de l'extraction de la cardinalité à partir de grands jeux de données. Vous pouvez pré-agréger ces jeux de données à l'aide des valeurs HLLSKETCH et les stocker dans des tables. Amazon Redshift peut extraire la cardinalité directement à partir des valeurs HLLSKETCH stockées sans accéder aux jeux de données sous-jacents.

Lors du traitement des esquisses HLL, Amazon Redshift effectue des optimisations qui minimisent l'empreinte mémoire de l'esquisse et maximisent la précision de la cardinalité extraite. Amazon Redshift utilise deux représentations pour les esquisses HLL : fragmentées et denses. Une HLLSKETCH démarre dans un format fragmenté. Sa taille augmente à mesure de l'insertion de nouvelles valeurs. Dès que la taille atteint celle de la représentation dense, Amazon Redshift convertit automatiquement l'esquisse du format fragmenté au format dense.

Amazon Redshift importe, exporte et imprime une HLLSKETCH au format JSON lorsque l'esquisse est dans un format fragmenté. Amazon Redshift importe, exporte et imprime une HLLSKETCH en

tant que chaîne Base64 lorsque l'esquisse est dans un format dense. Pour plus d'informations sur la commande UNLOAD, consultez Déchargement du type de données HLLSKETCH. Pour importer du texte ou des données CSV dans Amazon Redshift, utilisez la commande COPY. Pour de plus amples informations, veuillez consulter Chargement du type de données HLLSKETCH.

Pour plus d'informations sur les fonctions utilisées avec HyperLogLog, consultezHyperLogLog fonctions.

Rubriques

- Considérations
- Limites
- Exemples

Considérations

Cette rubrique décrit les détails d'utilisation HyperLogLog d'Amazon Redshift.

Voici les points à prendre en compte lors de l'utilisation HyperLogLog dans Amazon Redshift :

- Les HyperLogLog non-fonctions suivantes peuvent accepter une entrée de type HLLSKETCH ou des colonnes de type HLLSKETCH :
  - Fonction d’agrégation COUNT
  - Expressions conditionnelles COALESCE et NVL
  - Expressions CASE
- L'encodage pris en charge est RAW.
- Vous pouvez effectuer une opération UNLOAD sur une table avec des colonnes HLLSKETCH au format texte ou CSV. Vous pouvez utiliser les colonnes UNLOAD HLLSKETCH pour écrire des données HLLSKETCH. Amazon Redshift affiche les données au format JSON pour une représentation fragmentée ou au format Base64 pour une représentation dense. Pour plus d'informations sur la commande UNLOAD, consultez Déchargement du type de données HLLSKETCH.

  Ce qui suit montre le format utilisé pour une HyperLogLog esquisse éparse représentée au format JSON.

```
{"version":1,"logm":15,"sparse":{"indices":
[15099259,33107846,37891580,50065963],"values":[2,3,2,1]}}
```

- Vous pouvez importer du texte ou des données CSV dans Amazon Redshift à l'aide de la commande COPY. Pour de plus amples informations, veuillez consulter Chargement du type de données HLLSKETCH.
- L'encodage par défaut pour HLLSKETCH est RAW. Pour de plus amples informations, veuillez consulter encodages de compression.

Limites

Cette rubrique décrit les limites HyperLogLog d'Amazon Redshift.

Les limites d'utilisation HyperLogLog dans Amazon Redshift sont les suivantes :

- Les tables Amazon Redshift ne prennent pas en charge une colonne HLLSKETCH en tant que clé de tri ou clé de distribution.
- Amazon Redshift ne prend pas en charge les colonnes HLLSKETCH dans les clauses ORDER BY, GROUP BY ou DISTINCT.
- Vous pouvez décharger les colonnes UNLOAD HLLSKETCH uniquement au format texte ou CSV. Amazon Redshift écrit ensuite les données HLLSKETCH au format JSON ou Base64. Pour plus d'informations sur la commande UNLOAD, consultez UNLOAD.
- Amazon Redshift prend uniquement en charge les HyperLogLog esquisses dont la précision (valeur logm) est de 15.
- Les pilotes JDBC et ODBC ne prennent pas en charge le type de données HLLSKETCH. Par conséquent, l'ensemble de résultats utilise VARCHAR pour représenter les valeurs HLLSKETCH.
- Amazon Redshift Spectrum ne prend pas en charge nativement les données HLLSKETCH. Par conséquent, vous ne pouvez pas créer ni modifier de table externe avec une colonne HLLSKETCH.
- Les types de données pour les fonctions définies par l'utilisateur en Python (UDFs) ne prennent pas en charge le type de données HLLSKETCH. Pour plus d'informations sur Python UDFs, consultezPython scalaire UDFs.

Exemples

Cette section contient des exemples d'utilisation HyperLogLog avec Amazon Redshift.

Rubriques

- Exemple : renvoyer la cardinalité dans une sous-requête
- Exemple : renvoyer un type HLLSKETCH à partir d'esquisses combinées dans une sous-requête
- Exemple : renvoyer une HyperLogLog esquisse en combinant plusieurs esquisses
- Exemple : générer des HyperLogLog esquisses sur des données S3 à l'aide de tables externes

Exemple : renvoyer la cardinalité dans une sous-requête

L'exemple suivant renvoie la cardinalité de chaque esquisse d'une sous-requête pour une table nommée Sales (Ventes).

```
CREATE TABLE Sales (customer VARCHAR, country VARCHAR, amount BIGINT);
INSERT INTO Sales VALUES ('David Joe', 'Greece', 14.5),  ('David Joe', 'Greece',
 19.95), ('John Doe', 'USA', 29.95), ('John Doe', 'USA', 19.95), ('George Spanos',
 'Greece', 9.95), ('George Spanos', 'Greece', 2.95);
```

La requête suivante génère une esquisse HLL pour les clients de chaque pays et extrait la cardinalité. Cela montre des clients uniques de chaque pays.

```
SELECT hll_cardinality(sketch), country
FROM (SELECT hll_create_sketch(customer) AS sketch, country
        FROM Sales
        GROUP BY country) AS hll_subquery;

hll_cardinality | country
----------------+---------
            1   | USA
            2   | Greece
 ...
```

Exemple : renvoyer un type HLLSKETCH à partir d'esquisses combinées dans une sous-requête

L'exemple suivant renvoie un seul type HLLSKETCH qui représente la combinaison d'esquisses individuelles d'une sous-requête. Les esquisses sont combinées à l'aide de la fonction d'agrégation HLL_COMBINE.

```
SELECT hll_combine(sketch)
FROM (SELECT hll_create_sketch(customer) AS sketch
        FROM Sales
        GROUP BY country) AS hll_subquery

                                        hll_combine
-----------------------------------------------------------------------------------------------
 {"version":1,"logm":15,"sparse":{"indices":[29808639,35021072,47612452],"values":
[1,1,1]}}
(1 row)
```

Exemple : renvoyer une HyperLogLog esquisse en combinant plusieurs esquisses

Pour l'exemple suivant, supposons que la table `page-users` stocke des esquisses pré-agrégées pour chaque page visitée par les utilisateurs sur un site web donné. Chaque ligne de ce tableau contient une HyperLogLog esquisse représentant tous les utilisateurs IDs qui montrent les pages visitées.

```
page_users
-- +----------------+-------------+--------------+
-- | _PARTITIONTIME | page        | sketch |
-- +----------------+-------------+--------------+
-- | 2019-07-28     | homepage    | CHAQkAQYA... |
-- | 2019-07-28     | Product A   | CHAQxPnYB... |
-- +----------------+-------------+--------------+
```

L'exemple suivant réunit plusieurs esquisses pré-agrégées et génère une seule esquisse. Cette esquisse encapsule la cardinalité collective que chaque esquisse encapsule.

```
SELECT hll_combine(sketch) as sketch
```

```
FROM page_users
```

La sortie ressemble à ce qui suit.

```
-- +----------------------------------------+
-- | sketch |
-- +----------------------------------------+
-- | CHAQ3sGoCxgCIAuCB4iAIBgTIBgqgIAgAwY.... |
-- +----------------------------------------+
```

À la création d'une esquisse, vous pouvez utiliser la fonction HLL_CARDINALITY pour obtenir les valeurs collectives, comme illustré ci-dessous.

```
SELECT hll_cardinality(sketch)
FROM (
  SELECT
  hll_combine(sketch) as sketch
  FROM page_users
) AS hll_subquery
```

La sortie ressemble à ce qui suit.

```
-- +-------+
-- | count |
-- +-------+
-- | 54356 |
-- +-------+
```

Exemple : générer des HyperLogLog esquisses sur des données S3 à l'aide de tables externes

Les exemples suivants mettent en cache HyperLogLog des esquisses pour éviter d'accéder directement à Amazon S3 pour l'estimation de la cardinalité.

Vous pouvez préagréger et mettre en cache HyperLogLog des esquisses dans des tables externes définies pour contenir les données Amazon S3. Ce faisant, vous pouvez extraire des estimations de cardinalité sans accéder aux données

Par exemple, supposons que vous ayez déchargé un ensemble de fichiers texte délimités par des tabulations dans Amazon S3. Vous exécutez la requête suivante pour définir une table externe

nommée `sales` dans le schéma externe Amazon Redshift nommé `spectrum`. Dans cet exemple, le compartiment Amazon S3 se trouve dans l'est des États-Unis (Virginie du Nord) Région AWS.

```
create external table spectrum.sales(
salesid integer,
listid integer,
sellerid smallint,
buyerid smallint,
eventid integer,
dateid integer,
qtysold integer,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t' stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales/';
```

Supposons que vous souhaitiez calculer les personnes ayant acheté un article à des dates aléatoires. Pour ce faire, l'exemple suivant génère des croquis pour l'acheteur IDs pour chaque jour de l'année et enregistre le résultat dans le tableau Amazon Redshift. `hll_sales`

```
CREATE TABLE hll_sales AS
SELECT saletime, hll_create_sketch(buyerid) AS sketch
FROM spectrum.sales
GROUP BY saletime;

SELECT TOP 5 * FROM hll_sales;
```

La sortie ressemble à ce qui suit.

```
-- hll_sales

-- | saletime        | sketch
    |
-- +-----------------
+-------------------------------------------------------------------+
-- | 7/22/2008 8:30  | {"version":1,"logm":15,"sparse":{"indices":[9281416],"values":
[1]}}
-- | 2/19/2008 0:38  | {"version":1,"logm":15,"sparse":{"indices":[48735497],"values":
[3]}}
```

```
-- | 11/5/2008 4:49  | {"version":1,"logm":15,"sparse":{"indices":[27858661],"values":
[1]}}
-- | 10/27/2008 4:08 | {"version":1,"logm":15,"sparse":{"indices":[65295430],"values":
[2]}}
-- | 2/16/2008 9:37  | {"version":1,"logm":15,"sparse":{"indices":[56869618],"values":
[2]}}
-- +----------------
 +-----------------------------------------------------------------------+
```

La requête suivante indique le nombre estimé d'acheteurs distincts ayant acheté un article le vendredi suivant Thanksgiving en 2008.

```
SELECT hll_cardinality(hll_combine(sketch)) as distinct_buyers
FROM hll_sales
WHERE trunc(saletime) = '2008-11-28';
```

La sortie ressemble à ce qui suit.

```
distinct_buyers
---------------
386
```

Supposons que vous souhaitiez connaître le nombre d'utilisateurs ayant acheté un article à une certaine plage de dates. Par exemple, vous voudriez connaître les données entre le vendredi suivant Thanksgiving et le lundi. Pour ce faire, la requête suivante utilise la fonction d'agrégation `hll_combine`. Cette fonction évite de compter deux fois le nombre de personnes ayant acheté un article sur plusieurs jours de la plage sélectionnée.

```
SELECT hll_cardinality(hll_combine(sketch)) as distinct_buyers
FROM hll_sales
WHERE saletime BETWEEN '2008-11-28' AND '2008-12-01';
```

La sortie ressemble à ce qui suit.

```
distinct_buyers
---------------
1166
```

Pour conserver la `hll_sales` table up-to-date, exécutez la requête suivante à la fin de chaque journée. Cela génère un HyperLogLog croquis basé sur le nombre IDs d'acheteurs qui ont acheté un article aujourd'hui et l'ajoute au `hll_sales` tableau.

```
INSERT INTO hll_sales
SELECT saletime, hll_create_sketch(buyerid)
FROM spectrum.sales
WHERE TRUNC(saletime) = to_char(GETDATE(), 'YYYY-MM-DD')
GROUP BY saletime;
```

Requêtes entre bases de données

Cette rubrique décrit les requêtes entre bases de données, qui sont des requêtes qui fonctionnent sur plusieurs bases de données Amazon Redshift au sein d'un même cluster Amazon Redshift.

En utilisant des requêtes entre bases de données dans Amazon Redshift, vous pouvez interroger et écrire dans les bases de données d'un cluster Amazon Redshift. Avec les requêtes entre bases de données, vous pouvez interroger et écrire des données à partir de n'importe quelle base de données du cluster Amazon Redshift, quelle que soit la base de données à laquelle vous êtes connecté. Les requêtes entre bases de données évitent les copies de données et simplifient l’organisation de ces dernières pour prendre en charge plusieurs groupes professionnels à partir du même entrepôt des données.

Avec les requêtes entre bases de données, vous pouvez effectuer les opérations suivantes :

- Interroger plusieurs bases de données de votre cluster Amazon Redshift.

  Non seulement vous pouvez interroger à partir des bases de données auxquelles vous êtes connecté, mais vous pouvez également lire à partir de toutes les autres bases pour lesquelles vous avez des autorisations.

  Lorsque vous interrogez des objets de base de données sur d’autres bases de données non connectées, vous avez accès en lecture seule à ces objets de base de données. Vous pouvez utiliser des requêtes entre bases de données pour accéder aux données de l’une des bases de votre cluster Amazon Redshift sans avoir à vous connecter à cette dernière. Cela peut vous aider à interroger et à joindre rapidement et facilement des données réparties sur plusieurs bases de données de votre cluster Amazon Redshift.

  Vous pouvez également joindre des jeux de données à partir de plusieurs bases de données dans une seule requête et analyser les données à l’aide d’outils de Business Intelligence (BI) ou d’analytique. Vous pouvez continuer à configurer des contrôles d'accès en lecture granulaires au niveau des tables pour les utilisateurs à l'aide des commandes SQL Amazon Redshift standard. Ce faisant, vous pouvez vous assurer que les utilisateurs ne voient que les sous-jeux pertinents des données pour lesquelles ils ont des autorisations.
- Écrivez des données dans les bases de données de votre cluster Amazon Redshift. .

  Vous pouvez écrire à partir des bases de données auxquelles vous êtes connecté, mais également à partir de toute autre base de données pour laquelle vous êtes autorisé.

Lorsque vous disposez d'autorisations d'écriture sur les objets de base de données de toute autre base de données non connectée, vous pouvez utiliser des requêtes entre bases de données pour écrire des données depuis n'importe quelle base de données de votre cluster Amazon Redshift sans avoir à vous connecter à cette base de données spécifique. Cela peut vous aider à effectuer des opérations d'écriture complexes tout en joignant rapidement et facilement des données réparties sur plusieurs bases de données de votre cluster Amazon Redshift.

Vous pouvez également joindre des ensembles de données provenant de plusieurs bases de données en une seule requête et écrire les données à l'aide de différents outils extract-transform-load (ETL) ou d'analyse. Vous pouvez continuer à configurer des contrôles d'accès en écriture granulaires au niveau des tables pour les utilisateurs à l'aide des commandes SQL Amazon Redshift standard. Cela garantit que les utilisateurs ne voient que les sous-ensembles pertinents des données pour lesquels ils sont autorisés.

- Objets de requête.

  Vous pouvez interroger d'autres objets de base de données à l'aide de noms d'objets complets exprimés avec une notation en trois parties. Le chemin complet de tout objet de base de données a trois composants : le nom de la base de données, le schéma et le nom de l'objet. Vous pouvez accéder à n'importe quel objet depuis n'importe quelle autre base de données en utilisant la notation de chemin complet *database_name*.*schema_name*.*object_name*. Pour accéder à une colonne particulière, utilisez *database_name*.*schema_name*.*object_name*.*column_name*.

  Vous pouvez également créer un alias pour un schéma dans une autre base de données à l'aide de la notation de schéma externe. Ce schéma externe fait référence à une autre paire de base de données et de schéma. La requête peut accéder à l'autre objet de base de données en utilisant la notation de schéma externe *external_schema_name*.*object_name*.

  Dans la même requête en lecture seule, vous pouvez interroger divers objets de base de données, tels que des tables utilisateur, des vues standard, des vues matérialisées et des vues à liaison tardive provenant d'autres bases de données.

- Gérez les autorisations.

  Les utilisateurs disposant de privilèges d'accès aux objets de toutes les bases de données d'un cluster Amazon Redshift peuvent interroger ces objets et y écrire des données. Vous accordez des privilèges aux utilisateurs et aux groupes d'utilisateurs à l'aide de la commande GRANT. Vous pouvez également révoquer des privilèges à l'aide de la commande REVOKE lorsqu'un utilisateur n'a plus besoin d'accéder à des objets de base de données spécifiques.

- Travailler avec les métadonnées et les outils BI.

  Vous pouvez créer un schéma externe pour faire référence à un schéma dans une autre base de données Amazon Redshift au sein du même cluster Amazon Redshift. Pour plus d'informations, consultez la commande CREATE EXTERNAL SCHEMA.

  Une fois les références de schéma externes créées, Amazon Redshift affiche les tables sous le schéma de l'autre base de données dans SVV_EXTERNAL_TABLES et SVV_EXTERNAL_COLUMNS pour que les outils explorent les métadonnées.

  Pour intégrer une requête entre bases de données aux outils de BI, vous pouvez utiliser les vues système suivantes. Elles permettent d'afficher les métadonnées des objets dans les bases de données connectées et d'autres bases de données sur le cluster Amazon Redshift.

  Voici les vues système qui affichent les objets Amazon Redshift et les objets externes de toutes les bases de données de votre cluster Amazon Redshift :

  - SVV_ALL_COLUMNS
  - SVV_ALL_SCHEMAS
  - SVV_ALL_TABLES

  Voici les vues système qui affichent les objets Amazon Redshift de toutes les bases de données de votre cluster Amazon Redshift :

  - SVV_REDSHIFT_COLUMNS
  - SVV_REDSHIFT_DATABASES
  - SVV_REDSHIFT_FUNCTIONS
  - SVV_REDSHIFT_SCHEMAS
  - SVV_REDSHIFT_TABLES

Rubriques

- Considérations
- Limites
- Exemples de requêtes entre bases de données
- Utilisation de requêtes entre bases de données avec l'éditeur de requêtes

Considérations

Cette rubrique décrit les détails d'utilisation pour les requêtes entre bases de données dans Amazon Redshift.

Lorsque vous utilisez la fonction de requête entre bases de données dans Amazon Redshift, tenez compte des éléments suivants :

- Amazon Redshift prend en charge les requêtes entre bases de données sur tous les types de nœuds ra3 et espaces de noms sans serveur.
- Amazon Redshift prend en charge la jonction de données à partir de tables ou de vues dans une ou plusieurs bases de données du même cluster Amazon Redshift.
- Toutes les requêtes d'une transaction sur la base de données connectée lisent les données de la même façon que l'autre base de données avant la transaction. Cette approche permet d’assurer la cohérence transactionnelle des requêtes entre bases de données. Amazon Redshift prend en charge la cohérence transactionnelle pour les requêtes entre bases de données.
- Pour obtenir des métadonnées entre les bases de données, utilisez les vues de métadonnées SVV_ALL* et SVV_REDSHIFT*. Vous ne pouvez pas utiliser la notation en trois parties ou les schémas externes pour interroger des tables ou des vues de métadonnées entre bases de données sous information_schema et pg_catalog.

Limites

Cette rubrique décrit les limites relatives aux requêtes entre bases de données dans Amazon Redshift.

Lorsque vous travaillez avec la fonction de requête entre bases de données d’Amazon Redshift, tenez compte des limitations suivantes :

- Vous ne pouvez pas interroger les vues créées sur d’autres bases de données qui font référence à des objets d’une autre base de données.
- Vous ne pouvez pas écrire dans des schémas externes dans des objets de base de données sur d'autres bases de données non connectées.
- Vous pouvez uniquement créer des vues matérialisées et à liaison tardive sur des objets d’autres bases de données du cluster. Vous ne pouvez pas créer de vues régulières sur des objets d’autres bases de données du cluster.

- Amazon Redshift ne prend pas en charge les tables avec des privilèges au niveau des colonnes pour les requêtes entre bases de données.
- L'exécution de requêtes entre bases de données sur des tables avec des clés de tri entrelacées n'est pas prise en charge.

Exemples de requêtes entre bases de données

Cette rubrique contient des exemples d'utilisation de requêtes entre bases de données. Les requêtes entre bases de données sont des requêtes qui fonctionnent sur plusieurs bases de données au sein d'un même cluster Amazon Redshift.

Utilisez les exemples suivants pour apprendre à configurer une requête entre bases de données qui fait référence à une base de données Amazon Redshift.

Pour commencer, créez les bases de données db1 et db2 et les utilisateurs `user1` et `user2` dans votre cluster Amazon Redshift. Pour plus d'informations, consultez CREATE DATABASE et CREATE USER.

```
--As user1 on db1
CREATE DATABASE db1;

CREATE DATABASE db2;

CREATE USER user1 PASSWORD 'Redshift01';

CREATE USER user2 PASSWORD 'Redshift01';
```

Comme `user1` sur db1, créez une table, accordez des privilèges d'accès à `user2` et insérez des valeurs dans `table1`. Pour plus d'informations, consultez GRANT et INSERT.

```
--As user1 on db1
CREATE TABLE table1 (c1 int, c2 int, c3 int);

GRANT SELECT ON table1 TO user2;

INSERT INTO table1 VALUES (1,2,3),(4,5,6),(7,8,9);
```

En tant que `user2` sur db2, exécutez une requête entre base de données sur db2 en utilisant une notation en trois parties.

```
--As user2 on db2
SELECT * from db1.public.table1 ORDER BY c1;

c1 | c2  | c3
---+-----+----
1  |  2  | 3
4  |  5  | 6
7  |  8  | 9
(3 rows)
```

Accordez maintenant des privilèges d'écriture à`user2`, et insérez des valeurs `table1` dans in db1 as`user2`.

```
--As user1 on db1
GRANT INSERT ON table1 TO user2;
```

Ensuitedb2, `user2` exécutez une requête entre bases de données en db2 utilisant la notation en trois parties pour y insérer `table1` des données. db1

```
--As user2 on db2
INSERT INTO db1.public.table1 VALUES (10,11,12);
SELECT * from db1.public.table1 ORDER BY c1;

c1  | c2   | c3
----+------+----
1   |  2   | 3
4   |  5   | 6
7   |  8   | 9
10  |  11  | 12
(4 rows)
```

En tant que `user2` sur db2, créez un schéma externe et exécutez une requête entre bases de données sur db2 en utilisant une notation de schéma externe.

```
--As user2 on db2
CREATE EXTERNAL SCHEMA db1_public_sch
FROM REDSHIFT DATABASE 'db1' SCHEMA 'public';

SELECT * FROM db1_public_sch.table1 ORDER BY c1;

c1  | c2 | c3
```

```
----+----+----
1   | 2  | 3
4   | 5  | 6
7   | 8  | 9
10  | 11 | 12
(4 rows)
```

Pour créer différentes vues et accorder des autorisations, faites comme suit en tant que `user1` sur db1.

```
--As user1 on db1
CREATE VIEW regular_view AS SELECT c1 FROM table1;

GRANT SELECT ON regular_view TO user2;


CREATE MATERIALIZED VIEW mat_view AS SELECT c2 FROM table1;

GRANT SELECT ON mat_view TO user2;


CREATE VIEW late_bind_view AS SELECT c3 FROM public.table1 WITH NO SCHEMA BINDING;

GRANT SELECT ON late_bind_view TO user2;
```

En tant que `user2` sur db2, exécutez la requête entre bases de données suivante en utilisant une notation en trois parties pour afficher la vue particulière.

```
--As user2 on db2
SELECT * FROM db1.public.regular_view;

c1
----
1
4
7
10
(4 rows)

SELECT * FROM db1.public.mat_view;

c2
```

```
----
2
5
8
11
(4 rows)

SELECT * FROM db1.public.late_bind_view;

c3
----
3
6
9
12
(4 rows)
```

En tant que `user2` sur `db2`, exécutez la requête entre bases de données suivante en utilisant une notation de schéma externe pour interroger la vue à liaison tardive.

```
--As user2 on db2
SELECT * FROM db1_public_sch.late_bind_view;

c3
----
3
6
9
12
(4 rows)
```

En tant que `user2` sur `db2`, exécutez la commande suivante à l'aide de tables connectées dans une seule requête.

```
--As user2 on db2
CREATE TABLE table1 (a int, b int, c int);

INSERT INTO table1 VALUES (1,2,3), (4,5,6), (7,8,9);

SELECT a AS col_1, (db1.public.table1.c2 + b) AS sum_col2, (db1.public.table1.c3 + c)
 AS sum_col3 FROM db1.public.table1, table1 WHERE db1.public.table1.c1 = a;
col_1 | sum_col2 | sum_col3
```

```
------+----------+----------
1     | 4        | 6
4     | 10       | 12
7     | 16       | 18
(3 rows)
```

L'exemple suivant répertorie toutes les bases de données du cluster.

```
select database_name, database_owner, database_type
from svv_redshift_databases
where database_name in ('db1', 'db2');

 database_name | database_owner | database_type
---------------+----------------+---------------
 db1           |            100 | local
 db2           |            100 | local
(2 rows)
```

L'exemple suivant répertorie les schémas Amazon Redshift de toutes les bases de données du cluster.

```
select database_name, schema_name, schema_owner, schema_type
from svv_redshift_schemas
where database_name in ('db1', 'db2');

 database_name |    schema_name     | schema_owner | schema_type
---------------+--------------------+--------------+-------------
 db1           | pg_catalog         |            1 | local
 db1           | public             |            1 | local
 db1           | information_schema |            1 | local
 db2           | pg_catalog         |            1 | local
 db2           | public             |            1 | local
 db2           | information_schema |            1 | local
(6 rows)
```

L'exemple suivant répertorie les tables ou vues Amazon Redshift de toutes les bases de données du cluster.

```
select database_name, schema_name, table_name, table_type
from svv_redshift_tables
where database_name in ('db1', 'db2') and schema_name in ('public');
```

```
 database_name | schema_name |     table_name      | table_type
---------------+-------------+---------------------+------------
 db1           | public      | late_bind_view      | VIEW
 db1           | public      | mat_view            | VIEW
 db1           | public      | mv_tbl__mat_view__0 | TABLE
 db1           | public      | regular_view        | VIEW
 db1           | public      | table1              | TABLE
 db2           | public      | table2              | TABLE
(6 rows)
```

L'exemple suivant répertorie les schémas Amazon Redshift et externes de toutes les bases de données du cluster.

```
select database_name, schema_name, schema_owner, schema_type
from svv_all_schemas where database_name in ('db1', 'db2') ;

 database_name |    schema_name     | schema_owner | schema_type
---------------+--------------------+--------------+-------------
 db1           | pg_catalog         |            1 | local
 db1           | public             |            1 | local
 db1           | information_schema |            1 | local
 db2           | pg_catalog         |            1 | local
 db2           | public             |            1 | local
 db2           | information_schema |            1 | local
 db2           | db1_public_sch     |            1 | external
(7 rows)
```

L'exemple suivant répertorie les tables Amazon Redshift et externes de toutes les bases de données du cluster.

```
select database_name, schema_name, table_name, table_type
from svv_all_tables
where database_name in ('db1', 'db2') and schema_name in ('public');

 database_name | schema_name |     table_name      | table_type
---------------+-------------+---------------------+------------
 db1           | public      | regular_view        | VIEW
 db1           | public      | mv_tbl__mat_view__0 | TABLE
 db1           | public      | mat_view            | VIEW
 db1           | public      | late_bind_view      | VIEW
 db1           | public      | table1              | TABLE
```

```
 db2           | public      | table2              | TABLE
(6 rows)
```

Utilisation de requêtes entre bases de données avec l'éditeur de requêtes

Cette rubrique explique comment utiliser les requêtes entre bases de données avec l'éditeur de requêtes. Les requêtes entre bases de données sont des requêtes qui fonctionnent sur plusieurs bases de données au sein d'un même cluster Amazon Redshift.

Vous pouvez utiliser des requêtes entre bases de données pour accéder aux données de l'une des bases de votre cluster Amazon Redshift sans avoir à vous connecter à cette dernière. Lorsque vous exécutez des requêtes entre bases de données sur d'autres bases de données non connectées, vous n'avez accès en lecture et en écriture qu'à ces objets de base de données.

Vous pouvez interroger et écrire sur d'autres objets de base de données à l'aide de noms d'objets complets exprimés en notation en trois parties. Le chemin complet de tout objet de base de données a trois composants : le nom de la base de données, le schéma et le nom de l'objet. Par exemple : *database_name.schema_name.object_name*.

Pour utiliser des requêtes entre bases de données avec l'éditeur de requêtes v2

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Créez un cluster pour utiliser les requêtes entre bases de données dans l'éditeur de requêtes v2 Amazon Redshift. Pour plus d'informations, consultez Création d'un cluster du Guide de gestion Amazon Redshift.
3. Activez l'accès à l'éditeur de requêtes avec les autorisations appropriées. Pour plus d'informations, consultez Interrogation d'une base de données à l'aide de l'éditeur de requêtes v2 dans le Guide de gestion Amazon Redshift.
4. Dans le menu de navigation, choisissez Éditeur de requête v2, puis connectez-vous à une base de données dans votre cluster.

   Lorsque vous vous connectez à l'éditeur de requêtes v2 pour la première fois, Amazon Redshift affiche par défaut les ressources de la base de données connectée.
5. Choisissez les autres bases de données auxquelles vous avez accès pour afficher leurs objets de base de données. Pour afficher des objets, assurez-vous de disposer des autorisations

appropriées. Une fois la base de données choisie, Amazon Redshift affiche la liste des schémas associés.

Sélectionnez un schéma pour afficher la liste des objets de base de données associés.

> **Note**
>
> Amazon Redshift ne prend pas directement en charge les objets de catalogue de requêtes qui font partie des bases de données AWS Glue ou fédérées. Pour les interroger, créez d'abord des schémas externes faisant référence à ces sources de données externes dans chaque base de données.
> Les requêtes entre bases de données Amazon Redshift utilisant une notation en trois parties ne prennent pas en charge les tables de métadonnées sous les schémas `information_schema` et `pg_catalog`, car ces vues de métadonnées sont spécifiques à une base de données.

6. (Facultatif) Filtrez la liste des tables ou des vues pour le schéma que vous avez sélectionné.

Compatibilité avec Apache Iceberg pour Amazon Redshift

Vous pouvez enregistrer l'intégralité des clusters provisionnés par Amazon Redshift ou des espaces de noms sans serveur dans le afin de créer des catalogues qui partagent en toute sécurité AWS Glue Data Catalog les données en temps réel entre les comptes. AWS Vous pouvez accéder à ces catalogues depuis n'importe quel moteur de requête SQL prenant en charge l'API REST d'Apache Iceberg. AWS Lake Formation gère les autorisations pour les catalogues, vous permettant de gérer une seule copie des données avec un seul ensemble d'autorisations tout en tirant parti des fonctionnalités d'Amazon Redshift telles que les vues matérialisées et les intégrations sans ETL.

Tous les catalogues créés à partir de clusters provisionnés Amazon Redshift enregistrés et d'espaces de noms sans serveur sont automatiquement montés en AWS Glue Data Catalog tant que bases de données externes sur tous les clusters provisionnés et les groupes de travail sans serveur du même type sous le même compte. Région AWS Les catalogues créés dans le AWS Glue Data Catalog pour stocker des données dans Redshift Managed Storage (RMS) sont montés de la même manière que des bases de données externes. Une fois le montage terminé, vous pouvez vous connecter directement à ces bases de données et interroger les objets à l'aide de la notation `database@namespace-catalog.schema.table` en trois parties.

Régions dans lesquelles la compatibilité avec Apache Iceberg est disponible

La compatibilité d'Apache Iceberg avec Amazon Redshift est disponible dans les versions suivantes : Régions AWS

- USA Est (Virginie du Nord)
- USA Est (Ohio)
- USA Ouest (Californie du Nord)
- Asie-Pacifique (Hong Kong)
- Asie-Pacifique (Séoul) ;
- Asie-Pacifique (Singapour)
- Asie-Pacifique (Sydney)
- Asie-Pacifique (Tokyo)
- Canada (Centre)

- Europe (Francfort)
- Europe (Irlande)
- Europe (Londres)
- Europe (Stockholm)
- Amérique du Sud (São Paulo)

Considérations et limites relatives à l'utilisation des catalogues Amazon Redshift dans le AWS Glue Data Catalog

Lorsque vous utilisez les catalogues Amazon Redshift dans le AWS Glue Data Catalog, tenez compte des points suivants :

- Les entrepôts de données enregistrés AWS Glue Data Catalog suivent la syntaxe en trois parties pour accéder aux tables (`database@namespace-catalog.schema.table`). Par exemple, si vous avez enregistré un espace de noms Amazon Redshift nommé a, alimenté par une base de données nommée b, qui contenait un schéma nommé c, dans lequel se trouvait une table nommée d, vous devez sélectionner l'espace d en utilisant l'instruction suivante :

  ```
  SELECT * FROM b@a.c.d;
  ```

  Notez que la longueur combinée de la `database@namespace-catalog` partie de la syntaxe doit être inférieure ou égale à 127 caractères.
- Lorsque vous enregistrez un cluster ou un espace de noms dans le AWS Glue Data Catalog, Amazon Redshift enregistre toutes les bases de données et relations de ce cluster ou espace de noms.
- Vous pouvez enregistrer plusieurs clusters et espaces de noms Redshift dans le. AWS Glue Data Catalog
- L'enregistrement d'un cluster ou d'un espace de noms enregistre uniquement les schémas internes et les relations dans ce cluster ou cet espace de noms. Les éléments suivants ne sont pas enregistrés :
  - Schémas externes.
  - Tables extérieures. Notez que les vues à liaison tardive créées à partir de tables externes seront enregistrées.
  - Fonctions créées par l'utilisateur.

  - Procédures.
  - Tableaux auxquels sont jointes des politiques de sécurité ou de masquage dynamique des données au niveau des lignes.
  - Objets de base de données avec des noms en majuscules ou mixtes. Les tableaux contenant des colonnes majuscules ou mixtes ne sont pas enregistrés. Cela s'applique même lorsqu'il enable_case_sensitive_identifier est désactivé.
- Les autorisations de base de données Amazon Redshift, telles que les rôles accordés par le contrôle d'accès basé sur les rôles, ne sont pas transférées vers les catalogues du. AWS Glue Data Catalog AWS Lake Formation À utiliser pour configurer les autorisations pour AWS Glue Data Catalog. Pour plus d'informations sur l'utilisation de Lake Formation pour configurer les autorisations, consultez la section Configuration des autorisations pour les partages de données Amazon Redshift dans le manuel du développeur.AWS Lake Formation
- Lorsque vous créez un catalogue à partir d'un cluster enregistré ou d'un espace de noms sans serveur, un groupe de travail géré par Amazon Redshift est AWS Glue Data Catalog créé en utilisant les ressources informatiques d'Amazon Redshift pour répondre aux exigences de calcul lors de l'interrogation de ce catalogue. Vous pouvez consulter le groupe de travail géré dans la console Amazon Redshift Serverless et le gérer dans. AWS Glue
- Lorsque vous enregistrez un cluster suspendu, il ne AWS Glue Data Catalog sera pas monté en tant que catalogue tant que le cluster ne sera pas rétabli.
- Lorsque vous enregistrez un espace de noms sans serveur qui n'est pas utilisé activement, il AWS Glue Data Catalog ne sera pas monté en tant que catalogue tant que l'espace de noms ne sera pas réutilisé.
- Votre compte doit disposer d'un VPC par défaut pour créer un groupe de travail géré.

Exigences de la politique IAM pour accéder aux clusters et aux espaces de noms enregistrés dans le catalogue de données

Cette rubrique décrit les autorisations IAM requises pour enregistrer des clusters provisionnés et des espaces de noms sans serveur dans le catalogue de données et pour y accéder via Amazon Redshift.

Après avoir enregistré un cluster provisionné ou un espace de noms sans serveur dans le AWS Glue Data Catalog, vous devez disposer des autorisations suivantes pour découvrir la création et les modifications apportées au catalogue créé ultérieurement.

- `glue:GetCatalog`
- `glue:GetCatalogs`
- `lakeformation:GetDataAccess`

Ces autorisations sont incluses dans le rôle lié au service. `AmazonRedshiftServiceLinkedRolePolicy` Pour plus d'informations sur ce rôle, consultez la section Utilisation des rôles liés à un service pour Amazon Redshift dans le guide de gestion Amazon Redshift.

Enregistrement de clusters et d'espaces de noms Amazon Redshift auprès du AWS Glue Data Catalog

Vous pouvez ajouter des clusters provisionnés par Amazon Redshift et des espaces de noms sans serveur au pour y accéder AWS Glue Data Catalog à l'aide de l'API REST Apache Iceberg. Pour ce faire, vous devez enregistrer l'entrepôt de données Amazon Redshift à l' AWS Glue Data Catalog aide de la console Amazon Redshift AWS CLI ou créer un catalogue fédéré Amazon Redshift pour l'entrepôt utilisant. AWS Lake Formation

Les entrepôts de données Amazon Redshift enregistrés dans le catalogue de données agissent comme des partages de données producteurs. Les modifications que vous apportez aux clusters ou aux espaces de noms sans serveur à partir de leurs catalogues sont reflétées dans le cluster ou l'espace de noms dans Redshift, et vice versa.

Registering using the Amazon Redshift console

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Accédez à l'espace de noms sans serveur ou au cluster provisionné que vous souhaitez enregistrer et sélectionnez-le.
3. Sur la page de détails de l'espace de noms ou du cluster, sélectionnez S'inscrire dans le AWS Glue Data Catalog menu déroulant Actions.
4. Entrez l'ID du compte de destination sur lequel vous souhaitez enregistrer l'espace de noms ou le cluster, puis choisissez Enregistrer.
5. Si vous vous inscrivez depuis votre compte sur le même compte dans AWS Glue, vous serez redirigé directement vers la AWS Lake Formation console pour terminer le processus. Si vous vous inscrivez sur un autre compte, un lien apparaît qui vous amène à Lake Formation.

Registering using the AWS CLI

Pour enregistrer un cluster ou un espace de noms à l' AWS Glue Data Catalog aide du AWS CLI, utilisez la `register-namespace` commande avec les options suivantes :

- `namespace-identifier`: un objet avec l'identifiant unique du cluster ou de l'espace de noms que vous enregistrez. Cet objet est différent selon que vous enregistrez un cluster provisionné ou un espace de noms sans serveur. Éléments à prendre en compte :
  - Pour les clusters provisionnés, vous fournissez un `ProvisionedIdentifier` objet qui contient un `ClusterIdentifier` objet avec l'identifiant unique du cluster que vous enregistrez.
  - pour les espaces de noms sans serveur, vous fournissez un `ServerlessIdentifier` objet qui contient un `NamespaceIdentifier` objet avec l'identifiant unique de l'espace de noms que vous enregistrez, ainsi qu'un `WorkgroupIdentifier` objet avec l'identifiant unique du groupe de travail associé à cet espace de noms.
- `consumer-identifiers`: tableau contenant un seul élément, contenant l'identifiant unique du compte auprès duquel vous enregistrez le cluster ou l'espace de noms.

L'exemple suivant enregistre l'espace de noms `mySampleNamespace` sans serveur dans l'ID du compte. `012345678910`

```
aws redshift register-namespace /
--namespace-identifier {ServerlessIdentifier: {NamespaceIdentifer:
 mySampleNamespace, WorkgroupIdentifier: mySampleWorkgroup}} /
--consumer-identifiers [012345678910]
```

Annulation de l'enregistrement des clusters et des espaces de noms Amazon Redshift du AWS Glue Data Catalog

Vous pouvez annuler l'enregistrement d'un cluster provisionné ou d'un espace de noms sans serveur à l'aide de la console AWS Glue Data Catalog Amazon Redshift ou du. AWS CLI Les entrepôts de données dont vous vous désinscrivez ne AWS Glue Data Catalog sont supprimés qu'eux-mêmes. AWS Glue Data Catalog Les entrepôts font toujours partie de votre compte Amazon Redshift. De plus, le catalogue est conservé AWS Glue et doit être supprimé manuellement. Lorsque vous supprimez le catalogue AWS Glue, le groupe de travail géré qui lui est associé est également supprimé.

Deregistering using the Amazon Redshift console

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Accédez à l'espace de noms sans serveur ou au cluster provisionné dont vous souhaitez annuler l'enregistrement et sélectionnez-le.
3. Sur la page de détails de l'espace de noms ou du cluster, sélectionnez Désenregistrer dans le AWS Glue Data Catalog menu déroulant Actions. Cette option n'apparaît que si vous avez choisi un entrepôt de données déjà enregistré auprès du AWS Glue Data Catalog.
4. Entrez l'ID du compte propriétaire du catalogue contenant votre cluster ou votre espace de noms et choisissez Désenregistrer.

Deregistering using the AWS CLI

Pour annuler l'enregistrement d'un cluster ou d'un espace de noms à l' AWS Glue Data Catalog aide de AWS CLI, utilisez la `deregister-namespace` commande avec les options suivantes :

- `namespace-identifier`: un objet avec l'identifiant unique du cluster ou de l'espace de noms dont vous annulez l'enregistrement. Cet objet est différent selon que vous annulez l'enregistrement d'un cluster provisionné ou d'un espace de noms sans serveur. Éléments à prendre en compte :
  - Pour les clusters provisionnés, vous fournissez un `ProvisionedIdentifier` objet qui contient un `ClusterIdentifier` objet avec l'identifiant unique du cluster que vous annulez.
  - pour les espaces de noms sans serveur, vous fournissez un `ServerlessIdentifier` objet qui contient un `NamespaceIdentifier` objet avec l'identifiant unique de l'espace de noms que vous annulez, ainsi qu'un `WorkgroupIdentifier` objet avec l'identifiant unique du groupe de travail associé à cet espace de noms.
- `consumer-identifiers`: un tableau contenant un seul élément, contenant l'identifiant unique du compte dont vous annulez l'enregistrement du cluster ou de l'espace de noms.

L'exemple suivant désenregistre l'espace de noms `mySampleNamespace` sans serveur de l'ID de compte. `012345678910`

```
aws redshift deregister-namespace /
--namespace-identifier {ServerlessIdentifier: {NamespaceIdentifer:
 mySampleNamespace, WorkgroupIdentifier: mySampleWorkgroup}} /
```

```
--consumer-identifiers [012345678910]
```

Groupes de travail gérés

Lorsque vous enregistrez un cluster provisionné ou un espace de noms sans serveur dans le AWS Glue Data Catalog et que vous créez un catalogue à partir de celui-ci, vous AWS Glue créez un groupe de travail géré pour fournir des ressources de calcul à tout moteur de requête SQL accédant à ce catalogue. Par exemple, un Amazon Athena utilisateur interrogeant une table dans un catalogue verra ses exigences de calcul satisfaites par les ressources informatiques Amazon Redshift du groupe de travail géré. Les groupes de travail gérés utilisent les ressources de calcul Amazon Redshift, mais sont gérés dans. AWS Glue Votre compte doit disposer d'un VPC par défaut pour créer un groupe de travail géré.

Les groupes de travail gérés sont pris en compte dans le quota de groupes de travail Amazon Redshift Serverless, qui est de 25 par défaut. Pour plus d'informations, consultez Quotas et limites dans Amazon Redshift dans le Guide de gestion Amazon Redshift.

Vous pouvez consulter des informations générales, des indicateurs de performance et des informations de requête pour vos groupes de travail gérés dans la console Amazon Redshift Serverless. Pour plus d'informations, consultez la section Affichage des propriétés d'un groupe de travail dans le guide de gestion Amazon Redshift.

Consultation des catalogues enregistrés dans le AWS Glue Data Catalog

Une fois que vous avez enregistré un entrepôt de données Amazon Redshift dans AWS Glue Data Catalog et que vous avez défini les autorisations pour le catalogue obtenu AWS Lake Formation, le catalogue est automatiquement monté dans toutes les instances Amazon Redshift avec accès à l'entrepôt de données source sur le même compte et. Région AWS Vous pouvez ensuite interroger ce catalogue comme vous le feriez pour un cluster ou un groupe de travail local. Vous pouvez également interroger les catalogues enregistrés auprès du à l' AWS Glue Data Catalog aide de n'importe quel moteur SQL compatible avec l'API ouverte REST d'Apache Iceberg. Pour plus d'informations sur l'interrogation de catalogues à l' AWS Glue Data Catalog aide de l'API REST d'Apache Iceberg, consultez la section Accès au catalogue de données dans le guide du AWS Glue développeur. Pour plus d'informations sur l'API REST Apache Iceberg, consultez la spécification de l'API ouverte Apache Iceberg REST.

Pour interroger un catalogue, vous devez d'abord définir les autorisations pour le catalogue à l'aide de AWS Lake Formation. Pour plus d'informations sur la définition des autorisations pour les catalogues dans AWS Lake Formation, consultez la section Configuration des autorisations pour les partages de données Amazon Redshift dans le guide du développeur.AWS Lake Formation Vous avez également besoin d'un rôle IAM auquel est attachée la politique `AmazonRedshiftServiceLinkedRolePolicy` gérée. Pour plus d'informations sur les rôles liés à un service, consultez la section Utilisation des rôles liés à un service pour Amazon Redshift dans le guide de gestion Amazon Redshift.

Notez que les requêtes portant sur des catalogues doivent suivre la syntaxe en trois parties suivante pour accéder aux tables :

```
database@namespace.schema.table
```

Pour obtenir des informations générales sur l'interrogation des entrepôts de données Amazon Redshift, consultez la section Interrogation d'une base de données dans le guide de gestion Amazon Redshift.

Querying using the query editor v2

Après avoir défini les autorisations permettant à un compte d'accéder à un groupe de travail géré, ce groupe de travail géré apparaît dans le panneau d'arborescence situé sous la section des bases de données externes de votre base de données sans serveur. Vous pouvez interroger le groupe de travail géré de la même manière que vous interrogeriez un cluster interne approvisionné par Amazon Redshift ou un groupe de travail sans serveur, en utilisant le format de syntaxe en trois parties. `database@namespace/cluster.schema.table` Consultez l'exemple de déclaration suivant :

```
SELECT price FROM sales_db@mynamespace.sales_schema.inventory_table
```

Querying using the Data API

Vous pouvez interroger des groupes de travail gérés à l'aide de l'API Amazon Redshift Data de la même manière que vous interrogeriez un cluster interne approvisionné par Amazon Redshift ou un groupe de travail sans serveur, en transmettant le nom de ressource Amazon (ARN) du catalogue dans l'attribut approprié. `database` Prenons l'exemple suivant qui crée une table dans un catalogue.

```
aws redshift-data execute-statement –sql 'CREATE TABLE IF NOT EXISTS "dev@test-
rms-catalog".public.t1 (c1 INT, c2 VARCHAR(10));' –database arn:aws:glue:us-
east-1:550022730026:catalog/test-rms-catalog
```

Partage de données dans Amazon Redshift

Avec Amazon Redshift, vous pouvez partager des données en toute sécurité entre des clusters Amazon Redshift ou avec d'autres services. AWS Le partage de données vous permet de partager des données en temps réel, sans avoir à en créer une copie ou à les déplacer. Les administrateurs de bases de données et les ingénieurs de données peuvent utiliser le partage de données pour fournir un accès sécurisé en lecture seule aux données à des fins d'analyse, tout en gardant le contrôle sur les données. Les analystes de données, les professionnels de l'informatique décisionnelle et les scientifiques des données peuvent tirer parti des données partagées pour obtenir des informations sans les dupliquer ni les déplacer. Les cas d'utilisation courants incluent le partage de données avec des partenaires, la possibilité d'analyses interfonctionnelles et la facilitation de la démocratisation des données au sein d'une organisation. Les sections suivantes décrivent en détail la configuration et la gestion du partage de données dans Amazon Redshift.

Grâce au partage de données Amazon Redshift, vous pouvez partager en toute sécurité l'accès aux données en temps réel entre les clusters Amazon Redshift Régions AWS et les groupes de travail Comptes AWS, sans déplacer ni copier manuellement les données. Comme les données sont en ligne, tous les utilisateurs peuvent consulter les informations les plus complètes up-to-date et les plus cohérentes dans Amazon Redshift dès leur mise à jour.

Vous pouvez partager des données entre les clusters mis en service, les groupes de travail sans serveur, les zones de disponibilité, les Comptes AWS et les Régions AWS. Vous pouvez effectuer des partages entre différents types de cluster, mais aussi entre les clusters mis en service et les groupes de travail sans serveur.

Vous pouvez partager des objets de base de données en lecture et en écriture entre différents clusters Amazon Redshift ou groupes de travail Amazon Redshift Serverless au sein d'un Compte AWS même cluster ou de l'un à l'autre. Compte AWS Vous pouvez également lire et écrire des données dans toutes les régions. Vous pouvez accorder des autorisations telles que SELECT, INSERT et UPDATE pour différentes tables, et USAGE et CREATE pour différents schémas. Les données sont en ligne et disponibles dans tous les entrepôts dès qu’une transaction d’écriture est validée.

Cas d’utilisation du partage de données pour Amazon Redshift

Le partage de données Amazon Redshift est particulièrement utile dans les situations suivantes :

- Prise en charge de différents types d'applications essentielles à vos activités – utilisez un cluster central Extract-transform-load (ETL) qui partage des données avec plusieurs clusters de Business Intelligence (BI) ou d'analytique. Cette approche fournit l'isolation des charges de travail en lecture et la rétrofacturation pour les applications individuelles. Vous pouvez dimensionner et mettre à l'échelle votre calcul de charges de travail individuel en fonction des exigences spécifiques aux charges de travail en termes de prix et de performances.
- Activation de la collaboration entre groupes – permet une collaboration transparente entre les équipes et les groupes métier pour une analytique, une science des données et des analyses d'impact interproduits plus larges.
- Diffusion Data as a Service – partagez des données sous forme de services dans l'ensemble de votre organisation.
- Partage de données entre environnements – partagez des données entre les environnements de développement, de test et de production. Vous pouvez améliorer l'agilité des équipes en partageant des données selon différents niveaux de précision.
- Accès sous licence aux données dans Amazon Redshift — Répertoriez les ensembles de données Amazon Redshift dans AWS Data Exchange le catalogue que les clients peuvent trouver, auxquels ils peuvent s'abonner et consulter en quelques minutes.

Cas d'utilisation du partage de données et accès en écriture

Le partage de données pour les écritures présente plusieurs cas d'utilisation importants :

- Mettre à jour les données des sources commerciales sur le producteur : vous pouvez partager des données sous forme de service au sein de votre organisation, mais les consommateurs peuvent également effectuer des actions sur les données sources. Par exemple, ils peuvent communiquer des up-to-date valeurs rétrospectives ou accuser réception des données. Ce ne sont là que quelques cas d'utilisation commerciale possibles.
- Insérer des enregistrements supplémentaires sur le producteur — Les consommateurs peuvent ajouter des enregistrements aux données sources d'origine. Ils peuvent être marqués comme provenant du consommateur, si nécessaire.

Pour des informations spécifiques concernant la manière d'effectuer des opérations d'écriture sur un partage de données, consultez la section Partage de l'accès en écriture aux données.

Partage de données à différents niveaux dans Amazon Redshift

Avec Amazon Redshift, vous pouvez partager des données à différents niveaux. Ces niveaux incluent les bases de données, les schémas, les tables, les vues (y compris les vues standard, à liaison tardive et matérialisées) et les fonctions SQL définies par l'utilisateur (). UDFs Vous pouvez créer plusieurs unités de partage des données pour une base de données spécifique. Une unité de partage des données peut contenir des objets provenant de plusieurs schémas dans la base de données sur laquelle le partage est créé.

Grâce à cette flexibilité dans le partage des données, vous bénéficiez d'un contrôle d'accès affiné. Vous pouvez adapter ce contrôle aux différents utilisateurs et entreprises qui ont besoin d'accéder aux données Amazon Redshift.

Gestion de cohérence pour le partage de données dans Amazon Redshift

Amazon Redshift assure la cohérence des transactions sur tous les clusters et partages up-to-date de producteurs et de consommateurs, ainsi que des vues cohérentes des données avec tous les consommateurs.

Vous pouvez mettre à jour en continu les données du cluster producteur. Toutes les requêtes d'un cluster consommateur au sein d'une transaction affichent le même état des données partagées. Amazon Redshift ne prend pas en compte les données modifiées par une autre transaction sur le cluster producteur validé après le début de la transaction sur le cluster consommateur. Une fois que la modification des données est validée sur le cluster producteur, les nouvelles transactions sur le cluster consommateur peuvent immédiatement interroger les données mises à jour.

La forte cohérence élimine les risques liés aux rapports métier de faible fidélité qui peuvent contenir des résultats invalides lors du partage des données. Ce facteur est particulièrement important pour l'analyse financière ou lorsque les résultats peuvent être utilisés pour préparer des jeux de données qui sont utilisés pour entraîner des modèles de machine learning.

Considérations relatives au partage de données dans Amazon Redshift

Grâce au partage de données Amazon Redshift, vous pouvez partager en toute sécurité l'accès aux données en temps réel entre les clusters Amazon Redshift Régions AWS et les groupes de travail

Comptes AWS, sans déplacer ni copier manuellement les données. Auparavant, les objets des unités de partage des données étaient en lecture seule en toutes circonstances. L'écriture dans un objet d'une unité de partage des données est une nouvelle fonctionnalité. Les objets de unités de partage des données ne sont activés en écriture que lorsqu'un producteur accorde spécifiquement des privilèges d'écriture tels que INSERT ou CREATE sur des objets de l'unité de partage des données. En outre, pour le partage entre comptes, un producteur doit autoriser le partage de données pour les écritures et le consommateur doit associer des clusters et des groupes de travail spécifiques pour les écritures.

Cette section aborde les points à prendre en compte lors de l'utilisation du partage de données Amazon Redshift.

Considérations générales relatives au partage de données dans Amazon Redshift

Voici quelques considérations générales à prendre en compte lors de l'utilisation de partages de données dans Amazon Redshift :

- Base de données par défaut : lorsque vous lisez les données d'un partage de données, vous restez connecté à votre base de données de cluster locale. Pour plus d'informations sur la configuration et la lecture d'une base de données créée à partir d'un partage de données, consultez les sections Interrogation d'objets de partage de données et. Vues matérialisées sur des tables de lacs de données externes dans Amazon Redshift Spectrum
- Connexions : vous devez être connecté directement à une base de données de partage de données ou exécuter la commande USE pour écrire dans des partages de données. Vous pouvez également utiliser la notation en trois parties. La commande USE n'est pas prise en charge sur les tables externes.
- Performances — Les performances des requêtes sur les données partagées dépendent de la capacité de calcul des clusters consommateurs.
- Frais de transfert de données — Le partage de données entre régions inclut des frais supplémentaires de transfert de données entre régions.
  - Ces frais de transfert de données ne s'appliquent pas au sein d'une même région, mais uniquement entre les régions. Pour de plus amples informations, veuillez consulter Gestion du contrôle des coûts pour le partage de données entre régions.
  - Le consommateur est redevable de tous les frais de calcul et de transfert de données inter-région nécessaires pour interroger les données du producteur. Le producteur est facturé pour le

stockage sous-jacent des données dans son cluster mis en service ou son espace de noms sans serveur.

- Partage de données au sein des clusters et entre ceux-ci : vous n'avez besoin de partages de données que lorsque vous partagez des données entre différents clusters provisionnés par Amazon Redshift ou des groupes de travail sans serveur. Dans un même cluster, vous pouvez interroger une autre base de données en utilisant une notation simple en trois parties `database.schema.table`, à condition de disposer des autorisations nécessaires sur les objets de l'autre base de données.
- Découverte des métadonnées : lorsque vous êtes un consommateur connecté directement à une base de données de partage de données via les pilotes Redshift JDBC, ODBC ou Python, vous pouvez consulter les données du catalogue de la manière suivante :
  - Commandes SQL SHOW
  - Interrogation des vue et des tables information_schema
  - Interrogation des vues de métadonnées SVV
- Visibilité des autorisations — Les consommateurs peuvent voir les autorisations accordées aux partages de données via la commande SQL SHOW GRANTS.
- Gestion du chiffrement des clusters pour le partage des données : pour partager des données entre un cluster Compte AWS, le producteur et le cluster consommateur doivent être chiffrés.
  - Si les clusters producteurs et consommateurs et les espaces de noms sans serveur se trouvent dans le même compte, ils doivent avoir le même type de chiffrement (soit non chiffrés, soit tous deux chiffrés). Dans tous les autres cas, y compris les partages de données gérés par Lake Formation, le consommateur et le producteur doivent être chiffrés. ceci à des fins de sécurité. Cependant, ils ne sont pas tenus de partager la même clé de chiffrement.
  - Pour protéger les données en transit, toutes les données sont chiffrées en transit via le schéma de chiffrement du cluster producteur. Le cluster consommateur adopte ce schéma de chiffrement lorsque les données sont chargées. Le cluster consommateur fonctionne ensuite comme un cluster chiffré normal. Les communications entre le producteur et le consommateur sont également chiffrées à l'aide d'un schéma de clé partagée. Pour plus d'informations sur le chiffrement en transit, consultez Chiffrement en transit.

Considérations relatives au partage de données, de lecture et d'écriture dans Amazon Redshift

> **Note**
>
> Les écritures multi-entrepôts Amazon Redshift utilisant le partage de données ne sont prises en charge que sur le correctif 186 d'Amazon Redshift pour les clusters provisionnés sur la version 1.0.78881 ou supérieure, et pour les groupes de travail Amazon Redshift Serverless sur la version 1.0.78890 ou supérieure.

Voici quelques points à prendre en compte lors de l'utilisation de lectures et d'écritures de partage de données dans Amazon Redshift :

- Vous ne pouvez partager du SQL que par le UDFs biais de partages de données. Python et Lambda UDFs ne sont pas pris en charge.
- Si la base de données producteur a un classement spécifique, utilisez les mêmes paramètres de classement pour la base de données consommateur.
- Amazon Redshift ne prend pas en charge les fonctions SQL imbriquées définies par l'utilisateur sur les clusters producteur.
- Amazon Redshift ne prend pas en charge le partage de tables avec des clés de tri entrelacées et des vues qui font référence à des tables avec des clés de tri entrelacées.
- Amazon Redshift ne prend pas en charge l'accès à un objet d'unité de partage des données pour lequel un DDL se produit simultanément entre la préparation et l'exécution de l'accès.
- Amazon Redshift ne prend pas en charge le partage de procédures stockées via des unités de partage des données.
- Amazon Redshift ne prend pas en charge le partage de métadonnées, de vues système et de tables système.
- Type de calcul : vous devez utiliser des groupes de travail sans serveur, des clusters ra3.large, des clusters ra3.xlplus, des clusters ra3.4xl ou des clusters ra3.16xl pour utiliser cette fonctionnalité.
- Niveau d'isolation : le niveau d'isolation de votre base de données doit être une isolation instantanée afin de permettre aux autres groupes de travail sans serveur et aux clusters provisionnés d'y écrire.
- Requêtes et transactions à instructions multiples : les requêtes à instructions multiples en dehors d'un bloc de transactions ne sont actuellement pas prises en charge. Par conséquent, si vous

utilisez un éditeur de requêtes tel que dbeaver et que vous avez plusieurs requêtes d'écriture, vous devez les encapsuler dans une instruction de transaction BEGIN... END explicite.

Lorsque des instructions multicommandes sont utilisées en dehors des transactions, si la première commande est une écriture dans une base de données de production, les commandes d'écriture suivantes dans l'instruction ne sont autorisées que dans cette base de données de producteur. Si la première commande est une lecture, les commandes d'écriture suivantes ne sont autorisées que pour la base de données utilisée, si elles sont définies, sinon pour la base de données locale. Notez que les écritures d'une transaction ne sont prises en charge que dans une seule base de données.

- Dimensionnement des utilisateurs : les clusters de consommateurs doivent comporter au moins 64 tranches pour effectuer des écritures à l'aide du partage de données.
- Vues et vues matérialisées : vous ne pouvez pas créer, mettre à jour ou modifier des vues ou des vues matérialisées dans une base de données de partage de données.
- Sécurité — Vous ne pouvez pas associer ou supprimer des politiques de sécurité telles que le masquage dynamique des données (DDM) au niveau des colonnes (CLS), le niveau des lignes (RLS) et le masquage dynamique des données (DDM) aux objets de partage de données.
- Facilité de gestion — Les entrepôts de consommateurs ne peuvent pas ajouter d'objets de partage de données ou de vues faisant référence à des objets de partage de données à un autre partage de données. Les consommateurs ne peuvent pas non plus modifier ou supprimer un partage de données existant.
- Opérations de troncation : les écritures Datashare prennent en charge les troncs transactionnels pour les tables distantes. Cela est différent des troncs que vous exécutez localement sur un cluster, qui sont validés automatiquement. Pour plus d'informations sur la commande SQL, consultez TRUNCATE.

Considérations relatives au partage de données avec les tables de lacs de données dans Amazon Redshift

Les points suivants sont à prendre en compte lors de l'utilisation de tables de lacs de données dans Amazon Redshift :

- Le partage de données de tables de lacs de données ne prend pas en charge le chiffrement AWS KMS keys des compartiments Amazon S3 géré par le client. Vous pouvez l'utiliser Clés gérées par AWS pour le chiffrement. Pour plus d'informations, consultez Utilisation du chiffrement côté serveur avec des clés gérées par Amazon S3 (SSE-S3).

- Pour partager des tables de lacs de données à partir d'un AWS Glue catalogue chiffré, vous devez déléguer les AWS KMS opérations à un rôle IAM en suivant les instructions de la section Chiffrer votre catalogue de données.

Considérations relatives au partage de données avec AWS Lake Formation Amazon Redshift

Voici les considérations et les limites relatives au partage de données Amazon Redshift avec Lake Formation. Pour en savoir plus sur les considérations et les limites du partage de données, consultez Considérations relatives au partage de données dans Amazon Redshift. Pour plus d'informations sur les limites de Lake Formation, consultez Notes sur le travail avec les unités de partage des données d'Amazon Redshift dans Lake Formation.

- Le partage d'une unité de partage des données avec Lake Formation entre les régions n'est actuellement pas pris en charge.
- Si des filtres au niveau des colonnes sont définis pour un utilisateur sur une relation partagée, l'exécution d'une opération `SELECT *` ne renvoie que les colonnes auxquelles l'utilisateur a accès.
- Les filtres au niveau des cellules de Lake Formation ne sont pas pris en charge.
- Si vous avez créé et partagé une vue et ses tables avec Lake Formation, vous pouvez configurer des filtres pour gérer l'accès aux tables. Amazon Redshift applique les stratégies d'accès définies par Lake Formation lorsque les utilisateurs du cluster consommateur accèdent aux objets partagés. Lorsqu'un utilisateur accède à une vue partagée avec Lake Formation, Redshift n'applique que les stratégies d'accès définies sur la vue et non sur les tables contenues dans la vue. Toutefois, lorsque les utilisateurs accèdent directement à la table, Redshift applique les stratégies Lake Formation définies sur la table.
- Vous ne pouvez pas créer de vues matérialisées sur le consommateur à partir d'une table partagée si des filtres Lake Formation sont configurés sur cette table.
- L'administrateur Lake Formation doit disposer des autorisations d'administrateur de lac de données et des autorisations requises pour accepter une unité de partage des données.
- Le cluster producteur-consommateur doit être un RA3 cluster doté de la dernière version du cluster Amazon Redshift ou un groupe de travail sans serveur pour partager des partages de données via Lake Formation.
- Les clusters producteur et consommateur doivent être chiffrés.
- Les stratégies de contrôle d'accès Redshift de niveau lignes et colonnes mises en œuvre dans le cluster producteur ou le groupe de travail sont ignorées lorsque l'unité de partage des données est

partagée avec Lake Formation. L'administrateur Lake Formation doit configurer ces politiques dans Lake Formation. L'administrateur du cluster producteur ou du groupe de travail peut désactiver RLS pour une table à l'aide de la commande ALTER TABLE.

- Le partage de l'unité de partage des données via Lake Formation n'est disponible que pour les utilisateurs qui ont accès à la fois à Redshift et à Lake Formation.

Considérations relatives au partage de données avec AWS Data Exchange Amazon Redshift

Lorsque vous utilisez AWS Data Exchange Amazon Redshift, tenez compte des points suivants :

- Les producteurs et les consommateurs doivent utiliser les types d' RA3 instances pour utiliser les partages de données Amazon Redshift. Les producteurs doivent utiliser les types d' RA3 instances avec la dernière version du cluster Amazon Redshift.
- Les clusters producteur et consommateur doivent être chiffrés.
- Vous devez être enregistré en tant que AWS Data Exchange fournisseur pour mettre en vente des produits AWS Data Exchange, y compris les produits contenant des partages AWS Data Exchange de données. Pour plus d'informations, consultez Commencer en tant que fournisseur.
- Vous n'avez pas besoin d'être un AWS Data Exchange fournisseur enregistré pour rechercher des données Amazon Redshift, vous y abonner et les interroger. AWS Data Exchange
- Pour contrôler l'accès à vos données, créez des AWS Data Exchange partages de données avec le paramètre accessible au public activé. Pour modifier un partage de AWS Data Exchange données afin de désactiver le paramètre accessible au public, définissez la variable de session pour autoriser ALTER DATASHARE SET PUBLICACCESSIBLE FALSE. Pour de plus amples informations, veuillez consulter Notes d'utilisation d'ALTER DATASHARE.
- Les producteurs ne peuvent pas ajouter ou supprimer manuellement des consommateurs dans les AWS Data Exchange partages de données, car l'accès aux partages de données est accordé sur la base d'un abonnement actif à un AWS Data Exchange produit contenant le partage de données. AWS Data Exchange
- Les producteurs ne peuvent pas afficher les requêtes SQL exécutées par les consommateurs. Ils peuvent uniquement afficher les métadonnées, telles que le nombre de requêtes ou les requêtes des consommateurs d'objets, via des tables Amazon Redshift auxquelles seul le producteur peut accéder. Pour plus d'informations, consultez Surveillance et audit du partage des données dans Amazon Redshift.

- Nous vous recommandons de rendre vos unités de partage des données accessibles au public. Si vous ne le faites pas, les abonnés AWS Data Exchange appartenant à des clusters de consommateurs accessibles au public ne pourront pas utiliser votre partage de données.
- Nous vous recommandons de ne pas supprimer un partage de AWS Data Exchange données partagé avec d'autres personnes à Comptes AWS l'aide de l'instruction DROP DATASHARE. Si vous le faites, les Comptes AWS qui ont accès à l'unité de partage des données perdront l'accès. Cette action est irréversible. L'exécution de ce type de modification peut enfreindre les termes des produits de données dans AWS Data Exchange. Si vous souhaitez supprimer un partage de AWS Data Exchange données, consultez. Note d'utilisation de DROP DATASHARE
- Pour le partage de données entre régions, vous pouvez créer des partages AWS Data Exchange de données pour partager des données sous licence.
- Lors de la consommation de données provenant d'une autre région, le consommateur paie les frais de transfert de données entre régions de la région productrice vers la région consommatrice.

Autorisations que vous pouvez accorder aux unités de partages des données

Différents types d'objets et autorisations que vous pouvez leur accorder dans un contexte de partage de données.

Bases de données :

- CREATE
- USAGE
- ALTER
- DROP

Schémas :

- CREATE
- USAGE
- ALTER
- DROP

Tables :

- SELECT
- INSERT
- UPDATE
- DELETE
- TRUNCATE
- DROP
- ALTER
- REFERENCES

Fonctions :

- EXECUTE

Instructions SQL prises en charge pour le partage de données et écritures destinées aux consommateurs

Les instructions DDL (Data Definition Language) suivantes sont prises en charge pour le partage de données par écriture :

- ALTER TABLE RENAME TO
- MODIFIER LE TABLEAU, RENOMMER LA COLONNE EN
- MODIFIER LA ADD/DROP COLONNE DU TABLEAU
- MODIFIER LE SCHÉMA RENOMMÉ EN
- { CREATE | DROP } SCHEMA
- { CREATE | DROP | SHOW } TABLE
- CREATE TABLE table_name AS
- BEGIN | START TRANSACTION
- END | COMMIT | ROLLBACK
- TRUNCATE

Les instructions DML (Data Manipulation Language) suivantes sont prises en charge pour le partage de données par écriture :

- SELECT
- INSERT
- INSERT INTO SELECT
- UPDATE
- DELETE
- MERGE
- COPY without COMPUPDATE

Les instructions d'analyse suivantes sont prises en charge pour le partage de données avec des écritures :

- ANALYSER. Le consommateur exécute USER ANALYZE et envoie les statistiques au producteur.
- Analyse activée en CTAS/COPY exécutant sur le consommateur. Cela inclut le multiplexage pour les petits consommateurs.
- Exécution automatique de l'analyse sur le producteur après COPY.

Les déclarations d'autorisation suivantes sont prises en charge pour le partage de données avec des écritures :

- { GRANT | REVOKE } privilege_name ON OBJECT_TYPE object_name TO consumer_user
- AFFICHER LES SUBVENTIONS. Pour de plus amples informations, veuillez consulter AFFICHER LES SUBVENTIONS.

Des instructions SQL non prises en charge pour le partage de données s'adressent aux consommateurs

Les éléments suivants ne sont pas pris en charge :

- Requêtes à plusieurs instructions adressées aux entrepôts consommateur lors d’écritures vers les producteurs
- Requêtes à instructions multiples adressées aux entrepôts de consommateurs d'une autre base de données, si la commande précédente est une instruction de lecture.
- Références d'objet autres que les notations à trois points, telles que les notations à un point ou deux points, s'ils ne sont pas connectés à une base de données partagée.

- Requêtes de mise à l’échelle de la simultanéité avec écriture des consommateurs vers les producteurs
- Tâches de copie automatique avec écriture des consommateurs vers les producteurs.
- Tâches de streaming avec écriture des consommateurs vers les producteurs.
- Consommateurs créant des tables d’intégration zéro ETL sur les clusters producteur. Pour plus d’informations sur les intégrations zéro ETL, consultez Utilisation des intégrations zéro ETL.
- Écrire dans une table avec une clé de tri entrelacée via un partage de données.
- Écrire dans une procédure stockée par le biais d'un partage de données.
- Écrire dans une fonction SQL définie par l'utilisateur (UDF) via un partage de données. Il s'agit notamment de Nested, Python et Lambda UDFs.
- Instructions UPDATE, INSERT ou COPY sur les colonnes d'identité destinées aux entrepôts de consommateurs comportant plus de tranches de calcul que le producteur.
- Instructions MERGE sur des tables externes autres que RMS destinées aux entrepôts de consommateurs, lors de l'écriture aux producteurs.
- Instructions CREATE TABLE avec :
  - Expression DEFAULT définie sur le type de données VARBYTE. Le type de données VARBYTE ne peut pas être converti de façon implicite dans un autre type de données. Pour de plus amples informations, veuillez consulter Fonction CAST.
  - Instructions AS OF SELECT avec paramètre NULL destinées aux entrepôts de consommateurs, lors de l'écriture aux producteurs.
  - Paramètre LIKE pour les entrepôts de consommation, lorsque vous écrivez aux producteurs.

Régions AWS où le partage de données est disponible

Le tableau suivant répertorie la disponibilité des fonctionnalités de lecture et d'écriture pour le partage de données.

Région	Partage de données dans une même région	Partage de données entre régions	AWS Lake Formation partages de données régis
USA Est (Virginie du Nord) (us-east-1)	Oui	Oui	Oui

Région	Partage de données dans une même région	Partage de données entre régions	AWS Lake Formation partages de données régis
USA Est (Ohio) (us-east-2)	Oui	Oui	Oui
USA Ouest (Californie du Nord) (us-west-1)	Oui	Oui	Oui
USA Ouest (Oregon) (us-west-2)	Oui	Oui	Oui
Asie-Pacifique (Hong Kong) (ap-east-1)	Oui	Non	Non
Asie-Pacifique (Mumbai) (ap-south-1)	Oui	Oui	Oui
Asie-Pacifique (Hyderabad) (ap-south-2)	Oui	Non	Non
Asie-Pacifique (Tokyo) (ap-north east-1)	Oui	Oui	Oui
Asie-Pacifique (Singapour) (ap-south east-1)	Oui	Oui	Oui
Asie-Pacifique (Sydney) (ap-south east-2)	Oui	Oui	Oui
Asie-Pacifique (Jakarta) ; (ap-south east-3)	Oui	Non	Non

Région	Partage de données dans une même région	Partage de données entre régions	AWS Lake Formation partages de données régis
Asie-Pacifique (Melbourne) (ap-southeast-4)	Oui	Non	Non
Asie-Pacifique (Séoul) (ap-northeast-2)	Oui	Oui	Oui
Asie-Pacifique (Osaka) (ap-northeast-3)	Oui	Oui	Oui
Chine (Beijing) cn-north-1	Oui	Oui	Non
Chine (Ningxia) cn-northwest-1	Oui	Oui	Non
Afrique (Le Cap) (af-south-1)	Oui	Non	Non
Canada Ouest (Calgary) (ca-west-1)	Oui	Non	Non
Canada (Centre) (ca-central-1)	Oui	Oui	Oui
Europe (Francfort) (eu-central-1)	Oui	Oui	Oui
Europe (Zurich) (eu-central-2)	Oui	Non	Non
Europe (Irlande) (eu-west-1)	Oui	Oui	Oui

Région	Partage de données dans une même région	Partage de données entre régions	AWS Lake Formation partages de données régis
Europe (Londres) (eu-west-2)	Oui	Oui	Oui
Europe (Paris) (eu-west-3)	Oui	Oui	Oui
Europe (Milan) (eu-south-1)	Oui	Non	Non
Europe (Espagne) (eu-south-2)	Oui	Non	Non
Europe (Stockholm) (eu-north-1)	Oui	Oui	Oui
Moyen-Orient (Émirats arabes unis) (me-central-1)	Oui	Non	Non
Moyen-Orient (Bahreïn) (me-south-1)	Oui	Non	Non
Israël (Tel Aviv) (il-central-1)	Oui	Non	Non
Amérique du Sud (São Paulo) (sa-east-1)	Oui	Oui	Oui
AWS GovCloud (USA Est) (us-gov-east-1)	Oui	Non	Oui

Région	Partage de données dans une même région	Partage de données entre régions	AWS Lake Formation partages de données régis
AWS GovCloud (US-Ouest) (us-gov-west-1)	Oui	Non	Oui

Commencer à partager des données dans Amazon Redshift

Vous pouvez commencer à partager des données en suivant l'un des guides de cette section.

Rubriques

- Commencer à partager des données en lecture seule dans la console
- Commencer à partager des données en lecture seule avec l'interface SQL
- Commencer à utiliser les écritures dans plusieurs entrepôts à l'aide du partage de données dans Amazon Redshift
- Commencer à partager des données avec AWS CloudFormation Amazon Redshift

Commencer à partager des données en lecture seule dans la console

Avec Amazon Redshift, vous pouvez gérer le partage de données par écriture à l'aide de la console pour contrôler l'accès et gouverner les données sur les clusters et les comptes Amazon Redshift. AWS Les sections suivantes fournissent des step-by-step instructions sur la configuration et la gestion du partage de données par écriture à l'aide de la console Amazon Redshift.

Utilisez la console Amazon Redshift pour gérer les unités de partage des données créées sur votre compte ou partagées à partir d’autres comptes.

Vous avez besoin d’autorisations pour créer, modifier ou supprimer des unités de partage des données. Pour plus d'informations, consultez Gestion des autorisations pour un partage de données dans Amazon Redshift.

- Si vous êtes administrateur producteur, vous pouvez créer des partages de données, ajouter des consommateurs de données, ajouter des objets de partage de données, créer des bases de

données à partir de partages de données, modifier des partages de données ou supprimer des partages de données dans l'onglet CLUSTERS.

Dans le menu de navigation, accédez à l'onglet Clusters, choisissez un cluster dans la liste. Ensuite, effectuez l'une des actions suivantes :

- Cliquez sur l'onglet Unités de partage des données, choisissez une unité de partage des données dans la section Unités de partage des données créées dans mon cluster. Ensuite, effectuez l'une des actions suivantes :
  - Création d'un partage de données

    Lorsqu'une unité de partage des données est créé, vous pouvez ajouter des objets d'unité de partage des données ou des consommateurs de données. Pour plus d'informations, consultez Ajouter des objets de partage de données aux partages de données et Ajouter des consommateurs de données aux partages de données.
  - Modification des unités de partage des données créées dans votre compte
  - Supprimer un partage de données créé dans votre compte
- Choisissez Unités de partage des données et sélectionnez une unité de partage des données dans la section Unités de partage des données provenant d'autres clusters. Ensuite, effectuez l'une des actions suivantes :
  - Création d'un partage de données
  - Créer des bases de données à partir d'unités de partage des données
- Choisissez Databases (Bases de données) et sélectionnez une base de données dans la section Databases (Bases de données). Puis choisissez Créer une unité de partage des données. Pour plus d'informations, consultez Créer des bases de données à partir d'unités de partage des données.

> **Note**
>
> Pour afficher les bases de données et les objets des bases de données ou pour afficher les unités de partage des données du cluster, connectez-vous à une base de données. Pour plus d'informations, consultez Connexion à une base de données.

Connexion à une base de données

Avec Amazon Redshift, vous pouvez établir une connexion à votre cluster d'entrepôts de données et exécuter des requêtes SQL, charger des données ou effectuer des tâches administratives. La connexion à une base de données fait référence au processus de création d'un canal sécurisé entre une application ou un outil client et le cluster Amazon Redshift. Les sections suivantes fournissent des step-by-step instructions sur la manière de se connecter à une base de données Amazon Redshift.

Connectez-vous à une base de données pour afficher les bases de données et les objets des bases de données de ce cluster ou pour afficher les datashares.

Les informations d’identification utilisateur utilisées pour se connecter à une base de données spécifiée doivent disposer des autorisations nécessaires pour afficher toutes les unités de partage des données.

S’il n’y a pas de connexion locale, effectuez l’une des actions suivantes :

- Dans la page des détails de cluster, à partir de l'onglet Databases (Bases de données), dans la section Databases (Bases de données) ou Datashare objects (Objets d'unités de partage des données), choisissez Connect to database (Se connecter à la base de données) pour afficher les objets de base de données dans le cluster.
- Dans la page des détails de cluster, à partir de l’onglet Unités de partage des données, effectuez l’une des actions suivantes :
  - Dans la section Unités de partage des données provenant d’autres clusters), choisissez Connexion à la base de données pour afficher les unités de partage des données d’autres clusters.
  - Dans la section Datashares created in my cluster (Datashares créés dans mon cluster), choisissez Connect to database (Connexion à la base de données) pour afficher les datashares de votre cluster.
- Dans la fenêtre Connect to database (Connexion à la base de données), procédez de l’une des manières suivantes :
  - Si vous choisissez Create a new connection (Créer une connexion), sélectionnez AWS Secrets Manager pour utiliser un secret stocké afin d’authentifier l’accès à la connexion.

    Sinon, sélectionnez Temporary credentials (Informations d’identification temporaires) pour utiliser les informations d’identification de la base de données afin d’authentifier l’accès à la

connexion. Indiquez des valeurs pour Database name (Nom de base de données) et Database user (Utilisateur de la base de données).

Choisissez Se connecter.

- Choisissez Use a recent connection (Utiliser une connexion récente) pour vous connecter à une autre base de données pour laquelle vous disposez des autorisations nécessaires.

  Amazon Redshift établit automatiquement la connexion.

Une fois la connexion à la base de données établie, vous pouvez commencer à créer des partages de données, à interroger des partages de données ou à créer des bases de données à partir de partages de données.

Créer des unités de partage des données

Avec Amazon Redshift, vous pouvez partager des données en temps réel entre des clusters AWS ou des comptes Amazon Redshift à l'aide de partages de données. Un partage de données est un objet consommateur-producteur qui vous permet de partager des données en direct depuis votre cluster Amazon Redshift avec d'autres clusters ou comptes. AWS La création de partages de données permet un partage sécurisé des données tout en maintenant le contrôle de l'accès et en garantissant la conservation des données. up-to-date Les sections suivantes fournissent des informations détaillées sur la création de partages de données et l'ajout d'objets de base de données tels que des schémas, des tables et des vues pour partager des données en direct en toute sécurité.

Création d'un partage de données

Une unité de partage des données est un conteneur logique d’objets de base de données, d’autorisations et de consommateurs. Les consommateurs sont des clusters provisionnés par Amazon Redshift ou des espaces de noms Amazon Redshift Serverless dans votre compte, entre autres. Comptes AWS Chaque unité de partage des données est associée à la base de données dans laquelle elle a été créée, et seuls les objets de cette base de données peuvent être ajoutés. En tant qu'administrateur producteur, vous pouvez créer des partages de données sur la console et avec SQL en suivant l'une des procédures ci-dessous.

Console

Sur la console, vous pouvez créer des partages de données à partir des onglets Datashares de la page de détails du cluster ou de l'espace de noms. Une fois le partage de données créé, vous

pouvez créer des bases de données à partir du partage de données sur un consommateur en tant qu'administrateur du consommateur.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Sur la page de détails du cluster ou de l'espace de noms, depuis l'onglet Partage de données, dans la section Partage de données, connectez-vous à une base de données si vous n'êtes pas connecté à une base de données. Dans la section Datashare created in my account, sélectionnez Create datashare. La page Créer une unité de partage des données s’affiche.
4. Choisissez Créer une unité de partage des données. Vous pouvez uniquement créer une unité de partage des données à partir d’une base de données locale. Si vous ne vous êtes pas connecté à la base de données au préalable, la page Se connecter à la base de données s’affiche. Suivez les étapes de Connexion à une base de données pour vous connecter à une base de données. S’il existe une connexion récente, la page Créer une unité de partage des données s’affiche.
5. Dans la section Informations sur l’unité de partage des données, choisissez l’une des options suivantes :
   - Choisissez Datashare pour créer des partages de données afin de partager des données à des fins de lecture ou d'écriture entre différents entrepôts de données Amazon Redshift (clusters provisionnés ou points de terminaison sans serveur) ou dans un environnement identique ou différent. Compte AWS Comptes AWS
   - Choisissez le partage AWS Data Exchange de données pour créer des partages de données via lesquels octroyer des licences à vos données. AWS Data Exchange
6. Spécifiez les valeurs pour Nom de l’unité de partage des données, Nom de la base de données et Accessible publiquement. Lorsque vous modifiez le nom de la base de données, créez une nouvelle connexion à la base de données.
7. Ajoutez des objets à votre partage de données à l'aide des sections Autorisations étendues ou Autorisations directes. Pour ajouter des objets à un partage de données, consultez. Création d'un partage de données dans Amazon Redshift
8. Dans la section Consommateurs de données, vous pouvez choisir de publier sur Amazon Redshift ou de publier sur le AWS Glue Data Catalog, ce qui lance le processus de partage de données avec Lake Formation. Publier votre partage de données sur Amazon Redshift signifie

partager vos données avec un autre espace de noms ou un autre compte Amazon Redshift qui agit en tant que consommateur.

> **Note**
>
> Une fois l'unité de partage des données créée, vous ne pouvez pas modifier la configuration pour publier dans l'autre option.

9. Choisissez Créer une unité de partage des données.

SQL

La commande suivante crée une unité de partage des données :

```
CREATE DATASHARE salesshare;
```

Au moment de la création du partage de données, chaque partage de données est associé à une base de données. Seuls les objets de cette base de données peuvent être partagés dans ce datashare. Plusieurs datashares peuvent être créés sur la même base de données avec la même précision d'objets ou une précision différente. Il n'y a pas de limite au nombre d'unités de partage des données qu'un cluster peut créer. Vous pouvez également créer des datashares via la console Amazon Redshift. Pour de plus amples informations, veuillez consulter CREATE DATASHARE.

Vous pouvez également contrôler les restrictions de sécurité du partage de données lors de sa création. L'exemple suivant montre que le consommateur disposant d'un accès IP public est autorisé à lire le partage de données.

```
CREATE DATASHARE my_datashare [PUBLICACCESSIBLE = TRUE];
```

La définition de PUBLICACCESSIBLE = TRUE permet aux consommateurs d'interroger votre unité de partage des données à partir de clusters accessibles au public et de groupes de travail provisionnés. Oubliez ce paramètre ou définissez-le explicitement sur false si vous ne souhaitez pas l'autoriser.

Vous pouvez modifier les propriétés relatives au type de consommateurs après avoir créé un partage de données. Par exemple, vous pouvez définir que les clusters qui souhaitent consommer des données d'une unité de partage des données donné ne peuvent pas être

accessibles publiquement. Les requêtes provenant de clusters consommateur qui ne respectent pas les restrictions de sécurité spécifiées dans l'unité de partage des données sont rejetées au moment de l'exécution de la requête. Pour de plus amples informations, veuillez consulter ALTER DATASHARE.

Ajouter des objets de partage de données aux partages de données

Vous pouvez ajouter des objets de base de données de différents types sur la console et avec SQL en suivant l'une des procédures ci-dessous.

Console

Vous pouvez ajouter des objets à votre partage de données à l'aide des sections Autorisations étendues ou Autorisations directes. Sélectionnez Accorder des autorisations étendues ou Accorder des autorisations directes pour ajouter des objets. Sélectionnez le bouton Ajouter pour ajouter des objets. Une boîte de dialogue s'affiche. Procédez comme suit :

1. Si vous sélectionnez Accorder des autorisations étendues, la page Autorisations étendues d'octroi apparaît, dans laquelle vous pouvez accorder des autorisations étendues au niveau d'une base de données ou d'un schéma. Les partages de données dotés d'autorisations étendues disposent des autorisations spécifiées sur tous les objets actuels et futurs de la base de données ou du schéma. Pour plus de détails, voir,Autorisations étendues.

    1. Ensuite, sélectionnez Autorisations étendues de base de données pour accorder des autorisations étendues au niveau de la base de données. Lorsque vous accordez des autorisations étendues, elles s'appliquent à la base de données actuelle lors de la création du partage de données. Ces autorisations ne peuvent pas être accordées à des objets individuels et s'appliquent à la fois aux objets existants et aux nouveaux objets (schémas, tables, vues UDFs).
    2. Sélectionnez les autorisations étendues pour les schémas, les tables, les vues ou les fonctions définies par l'utilisateur. Cela signifie que tous les objets de la base de données disposent des autorisations sélectionnées accordées aux consommateurs. Sélectionnez Accorder pour terminer l'octroi des autorisations étendues à la base de données.
    3. Sélectionnez ensuite les autorisations étendues du schéma pour accorder des autorisations étendues au niveau du schéma. Lorsque vous accordez des autorisations étendues au schéma, tous les objets ajoutés au schéma disposent des autorisations de partage de données spécifiées.

4. Sélectionnez le schéma que vous souhaitez ajouter au partage de données dans la liste déroulante. Vous ne pouvez sélectionner qu'un seul schéma à la fois. Sélectionnez ensuite les autorisations directes que vous souhaitez accorder sur le schéma sélectionné.
5. Sélectionnez des autorisations étendues pour les objets de votre schéma tels que les tables, les vues et les fonctions définies par l'utilisateur. Les autorisations sont accordées sur tous les objets correspondants du schéma. Il peut s'agir d'objets existants ou d'objets ajoutés dans le futur. Lorsqu'elle est appliquée, vous ne pouvez pas supprimer une autorisation d'un objet sans révoquer les autorisations délimitées.
6. Sélectionnez Accorder pour terminer l'octroi des autorisations délimitées par le schéma.

2. Si vous sélectionnez Accorder des autorisations directes, la page Accorder des autorisations directes apparaît. Vous pouvez y accorder des autorisations directes au niveau de chaque objet, tel qu'un schéma, une table, une vue ou une fonction définie par l'utilisateur. Pour accorder des autorisations directes, vous devez d'abord ajouter les schémas appropriés au partage de données.

   1. Ensuite, sélectionnez Accorder des autorisations directes aux schémas pour appliquer des autorisations directes à un schéma spécifique. Sélectionnez ensuite les autorisations de schéma pour vos objets de schéma tels que les tables, les vues et les fonctions définies par l'utilisateur, puis sélectionnez le schéma que vous souhaitez ajouter au partage de données. Sélectionnez Grant pour terminer l'ajout du schéma au partage de données.
   2. Une fois qu'un schéma a été ajouté à votre partage de données, vous pouvez ajouter des autorisations directes pour vos objets de schéma. Sélectionnez à nouveau Accorder des autorisations directes. La page Accorder des autorisations directes apparaît. Accédez ensuite aux onglets d'autorisations directes pour les objets du schéma.
   3. Sélectionnez Accorder des autorisations directes aux tables et aux vues pour accorder des autorisations directes au niveau des objets sur ces objets. Sélectionnez les autorisations directes requises et les objets requis dans la liste. Utilisez le champ de recherche pour trouver des objets de partage de données. Sélectionnez Grant pour terminer l'ajout de tables et de vues au partage de données.
   4. Sélectionnez Accorder des autorisations directes aux fonctions définies par l'utilisateur pour accorder des autorisations directes au niveau de l'objet sur les fonctions définies par l'utilisateur. Sélectionnez les autorisations directes requises et les objets requis dans la liste. Utilisez le champ de recherche pour trouver des objets de partage de données. Sélectionnez Grant pour terminer l'ajout de fonctions définies par l'utilisateur au partage de données.

3. Vous pouvez également choisir d'ajouter de futurs objets ou non. Lorsque vous choisissez d'inclure les objets d'unité de partage des données ajoutés au schéma, cela signifie que les objets ajoutés au schéma sont automatiquement ajoutés à l'unité de partage des données.
4. Choisissez Ajouter pour terminer la section et ajouter les objets. Ceux-ci sont répertoriés sous Objets d'unité de partage des données.
5. Après avoir ajouté des objets, vous pouvez sélectionner des objets individuels et modifier leurs autorisations. Si vous sélectionnez un schéma, une boîte de dialogue apparaît pour vous demander si vous souhaitez ajouter des autorisations étendues. Ainsi, chaque objet existant ou ajouté au schéma dispose d'un ensemble d'autorisations présélectionnées, adaptées au type d'objet. Par exemple, l'administrateur peut définir que toutes les tables ajoutées disposent des autorisations SELECT et UPDATE.
6. Tous les objets de partage de données sont répertoriés dans les sections Autorisations étendues ou Autorisations directes.
7. Dans la section Consommateurs de données, vous pouvez ajouter des espaces de noms ou ajouter AWS des comptes en tant que consommateurs du partage de données.
8. Sélectionnez Créer une unité de partage des données pour enregistrer vos modifications.

Une fois que vous avez créé l'unité de partage des données, il apparaît dans la liste sous Unités de partage des données créées dans mon espace de noms. Si vous choisissez une unité de partage des données dans la liste, vous pouvez afficher ses consommateurs, ses objets et d'autres propriétés.

SQL

Avec SQL, le propriétaire du partage de données doit accorder l'autorisation USAGE aux schémas qu'il souhaite ajouter au partage de données. Le GRANT est utilisé pour accorder diverses actions sur le schéma, notamment CREATE et USAGE. Les schémas contiennent des objets partagés :

```
CREATE SCHEMA myshared_schema1;
CREATE SCHEMA myshared_schema2;

GRANT USAGE ON SCHEMA myshared_schema1 TO DATASHARE my_datashare;
GRANT CREATE, USAGE ON SCHEMA myshared_schema2 TO DATASHARE my_datashare;
```

L'administrateur peut également continuer d'exécuter les commandes ALTER pour ajouter un schéma à l'unité de partage des données. Seules les autorisations USAGE sont accordées lorsqu'un schéma est ajouté de cette façon.

```
ALTER DATASHARE my_datashare ADD SCHEMA myshared_schema1;
```

Une fois que l'administrateur a ajouté des schémas, il peut accorder des autorisations d'unité de partage des données sur les objets du schéma. Il peut s'agit d'autorisations en lecture ou en écriture. L'exemple GRANT ALL montre comment accorder toutes les autorisations.

```
GRANT SELECT, INSERT ON TABLE myshared_schema1.table1, myshared_schema1.table2,
 myshared_schema2.table1
TO DATASHARE my_datashare;

GRANT ALL ON TABLE myshared_schema1.table4 TO DATASHARE my_datashare;
```

Vous pouvez continuer d'exécuter des commandes comme ALTER DATASHARE pour ajouter des tables. Dans ce cas, seules les autorisations SELECT sont accordées aux objets ajoutés.

```
ALTER DATASHARE my_datashare ADD TABLE myshared_schema1.table1,
 myshared_schema1.table2, myshared_schema2.table1;
```

Vous pouvez accorder des autorisations étendues à un partage de données sur tous les objets d'un type donné au sein d'une base de données ou d'un schéma. Les partages de données dotés d'autorisations étendues disposent des autorisations spécifiées sur tous les objets actuels et futurs de la base de données ou du schéma.

Vous pouvez consulter l'étendue des autorisations délimitées au niveau de la base de données dans. SVV_DATABASE_PRIVILEGES Vous pouvez consulter l'étendue des autorisations délimitées au niveau du schéma dans. SVV_SCHEMA_PRIVILEGES

Voici la syntaxe permettant d'accorder des autorisations étendues aux partages de données. Pour plus d'informations sur les autorisations délimitées, consultezAutorisations étendues.

```
GRANT { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }FOR SCHEMAS IN
DATABASE db_name
TO DATASHARE { datashare_name}

GRANT { { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [PRIVILEGES] } }FOR TABLES IN
```

```
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO DATASHARE { datashare_name}

GRANT { EXECUTE | ALL [ PRIVILEGES ] }FOR FUNCTIONS IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO DATASHARE { datashare_name}
```

Ajouter des consommateurs de données aux partages de données

Vous pouvez ajouter un ou plusieurs consommateurs de données aux unités de partage des données. Les consommateurs de données peuvent être des espaces de noms identifiant de manière unique les clusters Amazon Redshift ou. Comptes AWS

Vous devez explicitement choisir de désactiver ou d'activer le partage de votre unité de partage des données vers des clusters avec accès public.

- Choisissez Ajouter des espaces de noms au partage de données. Les espaces de noms sont un identifiant global unique (GUID) pour le cluster Amazon Redshift.
- Choisissez Add Comptes AWS(Ajouter un Comptes AWS) pour l'unité de partage des données. La personne spécifiée Comptes AWS doit disposer d'autorisations d'accès au partage de données.

Autoriser ou supprimer l'autorisation des unités de partage des données

En tant qu'administrateur producteur, choisissez les consommateurs de données à autoriser à accéder aux partages de données ou auxquels vous souhaitez retirer leur autorisation. Les consommateurs de données autorisés reçoivent des notifications pour prendre des mesures sur les unités de partage des données. Si vous ajoutez un espace de noms en tant que consommateur de données, vous n'avez pas à effectuer d'autorisation.

Prérequis : pour autoriser ou retirer l'autorisation pour l'unité de partage des données, il doit y avoir au moins un consommateur de données ajouté à l'unité de partage des données.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Datashares. La page de liste d'unités de partage des données s'affiche.
3. Choisissez Dans mon compte.

4. Dans la section Unités de partage des données de mon compte, effectuez l'une des opérations suivantes :
   - Choisissez un ou plusieurs clusters consommateur que vous souhaitez autoriser. La page Authorize data consumers (Autoriser les consommateurs de données) s'affiche. Choisissez ensuite Authorize (Autoriser).

     Si vous avez choisi Publier sur AWS Glue Data Catalog au moment de créer l'unité de partage des données, vous ne pouvez accorder l'autorisation de l'unité de partage des données qu'à un compte Lake Formation.

     Pour le AWS Data Exchange partage de données, vous ne pouvez autoriser qu'un seul partage de données à la fois.

     Lorsque vous autorisez un partage de AWS Data Exchange données, vous partagez le partage de données avec le AWS Data Exchange service et vous autorisez AWS Data Exchange à gérer l'accès au partage de données en votre nom. AWS Data Exchange permet l'accès aux consommateurs en ajoutant des comptes consommateurs en tant que consommateurs de données au AWS Data Exchange partage de données lorsqu'ils s'abonnent aux produits. AWS Data Exchange n'a pas d'accès en lecture au partage de données.
   - Choisissez un ou plusieurs clusters consommateur pour lesquels vous souhaitez supprimer l'autorisation. Puis choisissez Remove authorization (Supprimer l'autorisation).

Une fois que les consommateurs de données sont autorisés, ils peuvent accéder aux objets d'unité de partage des données et créer une base de données consommateur pour interroger les données.

Une fois l'autorisation supprimée, les consommateurs de données perdent immédiatement l'accès à l'unité de partage des données.

Gestion d'unités de partage des données provenant d'autres comptes en tant que consommateur

Supprimer l'association d'unité de partage des données des consommateurs de données

En tant qu'administrateur de consommateurs, vous pouvez supprimer l'association de partages de données aux consommateurs de données.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/

2. Dans le menu de navigation, choisissez Datashares. La page de liste d'unités de partage des données s'affiche.
3. Sélectionnez À partir d'autres comptes.
4. Dans la section Datashares from other accounts (Datashares provenant d'autres comptes), choisissez le datashare duquel supprimer l'association des consommateurs de données.
5. Dans la section Data consumers (Consommateurs de données), sélectionnez un ou plusieurs consommateurs de données desquels supprimer l'association. Puis choisissez Remove association (Supprimer l'association).
6. Lorsque la page Remove association (Supprimer l'association) s'affiche, choisissez Remove association (Supprimer l'association).

Une fois l'association supprimée, les consommateurs de données perdront l'accès à l'unité de partage des données. Vous pouvez modifier l'association de consommateurs de données à tout moment.

Refus d'unités de partage des données

En tant qu'administrateur client, vous pouvez rejeter tout partage de données dont l'état est disponible ou actif. Après avoir rejeté une unité de partage des données, les utilisateurs du cluster consommateur perdent l'accès à l'unité de partage des données. Amazon Redshift ne renvoie pas l'unité de partage des données rejetée si vous appelez l'opération d'API `DescribeDataSharesForConsumer`. Si l'administrateur du producteur exécute l'opération `DescribeDataSharesForProducer` API, il constatera que le partage de données a été rejeté. Lorsqu'un partage de données est rejeté, l'administrateur du producteur peut à nouveau autoriser le partage de données à un cluster de consommateurs, et l'administrateur du consommateur peut choisir d'associer son AWS compte au partage de données ou de le rejeter.

Si votre AWS compte est associé à un partage de données et qu'il existe une association en attente à un partage de données géré par Lake Formation, le rejet de l'association de partage de données gérée par Lake Formation entraîne également le rejet du partage de données d'origine. Pour rejeter une association spécifique, l'administrateur du producteur peut supprimer l'autorisation d'un partage de données spécifié. Cette action n'affecte pas les autres unités de partage des données.

Pour rejeter un partage de données, utilisez la AWS console, l'opération `RejectDataShare` API ou `reject-datashare` dans le. AWS CLI

Pour rejeter un partage de données à l'aide de la AWS console :

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, sélectionnez Unités de partage des données.
3. Sélectionnez À partir d’autres comptes.
4. Dans la section Datashares from other accounts (Datashares provenant d'autres comptes), choisissez le datashare que vous souhaitez refuser. Lorsque la page Refuser l’unité de partage des données s’affiche, choisissez Refuser.

Une fois que vous avez refusé les unités de partage des données, vous ne pouvez pas revenir en arrière. Amazon Redshift supprime les unités de partage des données de la liste. Pour voir à nouveau l’unité de partage des données, l’administrateur du producteur doit l’autoriser à nouveau.

Gestion d’unités de partage des données existantes

Avec Amazon Redshift, vous pouvez gérer les partages de données existants afin de contrôler l'accès à vos données dans un cluster Amazon Redshift. Les sections suivantes fournissent des step-by-step conseils sur la gestion des partages de données dans votre environnement Amazon Redshift.

Affichage des unités de partage des données

Afficher les jeux de données à partir de l'onglet DATASHARES ou CLUSTERS.

- Utilisez l’onglet DATASHARES pour répertorier les unités de partage des données de votre compte ou provenant d’autres comptes.
  - Pour afficher les unités de partage des données créées dans votre compte, choisissez Dans mon compte, puis choisissez l’unité de partage des données que vous souhaitez afficher.
  - Pour afficher les unités de partage des données partagées à partir d’autres comptes, choisissez Provenant d’autres comptes, puis choisissez l’unité de partage des données que vous souhaitez afficher.
- Utilisez l’onglet CLUSTERS pour répertorier les unités de partage des données de votre cluster ou provenant d’autres clusters.

  Connectez-vous à une base de données. Pour plus d'informations, consultez Connexion à une base de données.

Ensuite, choisissez un datashare dans la section Datashares from other clusters (Datashares provenant d'autres clusters) ou Datashares created in my cluster (Datashares créés dans mon cluster) pour afficher ses détails.

Supprimer des objets d’unité de partage des données des unités de partage des données

Vous pouvez supprimer un ou plusieurs objets d’une unité de partage des données à l’aide de la procédure suivante.

Pour supprimer un ou plusieurs objets d’une unité de partage des données

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Choisissez Datashares.
4. Dans la section Datashares created in my account (Datashares créés sur mon compte), choisissez Connect to database (Connexion à la base de données). Pour plus d'informations, consultez Connexion à une base de données.
5. Choisissez l’unité de partage des données à modifier, puis choisissez Modifier. La page de détails de l’unité de partage des données s’affiche.
6. Pour supprimer un ou plusieurs objets d’unité de partage des données d’une unité de partage des données, effectuez l’une des actions suivantes :

   - Pour supprimer des schémas d’une unité de partage des données, sélectionnez un ou plusieurs schémas. Puis choisissez Remove (Supprimer). Amazon Redshift supprime les schémas spécifiés et tous les objets des schémas spécifiés de l’unité de partage des données.
   - Pour supprimer des tables et des vues de l’unité de partage des données, sélectionnez une ou plusieurs tables et vues. Puis choisissez Remove (Supprimer). Sinon, choisissez Remove by schema (Supprimer par schéma) pour supprimer toutes les tables et vues des schémas spécifiés.
   - Pour supprimer des fonctions définies par l’utilisateur de l’unité de partage des données, sélectionnez une ou plusieurs fonctions définies par l’utilisateur. Puis choisissez Remove (Supprimer). Sinon, choisissez Remove by schema (Supprimer par schéma) pour supprimer toutes les fonctions définies par l’utilisateur dans les schémas spécifiés.

Supprimer des consommateurs de données des unités de partage des données

Vous pouvez supprimer un ou plusieurs consommateurs de données d'une unité de partage des données. Les consommateurs de données peuvent être des espaces de noms qui identifient de manière unique les clusters AWS ou les comptes Amazon Redshift.

Choisissez un ou plusieurs consommateurs de données dans l'espace de noms IDs ou dans le AWS compte, puis choisissez Supprimer.

Amazon Redshift supprime les consommateurs de données spécifiés de l'unité de partage des données. Ils perdent immédiatement l'accès au partage de données.

Modification des unités de partage des données créées dans votre compte

Modifier les datashares créés dans votre compte à l'aide de la console. Connectez-vous d'abord à une base de données pour afficher la liste des unités de partage des données créés dans votre compte.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Choisissez Datashares.
4. Dans la section Datashares created in my account (Datashares créés sur mon compte), choisissez Connect to database (Connexion à la base de données). Pour plus d'informations, consultez Connexion à une base de données.
5. Choisissez l'unité de partage des données à modifier, puis choisissez Modifier. La page de détails de l'unité de partage des données s'affiche.
6. Effectuez des modifications dans la section Objets d'unité de partage des données ou Consommateurs de données.

   > **Note**
   >
   > Si vous avez choisi de publier votre partage de données sur le AWS Glue Data Catalog, vous ne pouvez pas modifier la configuration pour publier le partage de données sur d'autres comptes Amazon Redshift.

7. Sélectionnez Enregistrer les modifications.

Amazon Redshift met à jour votre unité de partage des données avec les modifications.

Supprimer un partage de données créé dans votre compte

Supprime les datashares créés dans votre compte à l'aide de la console. Connectez-vous d'abord à une base de données pour afficher la liste des unités de partage des données créés dans votre compte.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Choisissez Datashares. La liste d'unités de partage des données s'affiche.
4. Dans la section Datashares created in my account (Datashares créés sur mon compte), choisissez Connect to database (Connexion à la base de données). Pour plus d'informations, consultez Connexion à une base de données.
5. Choisissez une ou plusieurs unités de partage des données à supprimer, puis sélectionnez Supprimer. La page Supprimer des unités de partage des données s'affiche.

   La suppression d'une unité de partage des données partagée avec Lake Formation ne supprime pas automatiquement les autorisations associées dans Lake Formation. Pour les supprimer, accédez à la console Lake Formation.
6. Saisissez Supprimer pour confirmer la suppression des unités de partage des données spécifiées.
7. Sélectionnez Delete (Supprimer).

Une fois les unités de partage des données supprimées, les consommateurs d'unités de partage des données perdent l'accès aux unités de partage des données.

Interrogation d'unités de partage des données

Avec Amazon Redshift, vous pouvez interroger des données provenant de clusters de producteurs dans des partages de données afin d'accéder en toute sécurité aux données en temps réel sans les copier ni les transférer. Les sections suivantes décrivent en détail la configuration et l'interrogation des partages de données dans votre environnement Amazon Redshift.

Créer des bases de données à partir d’unités de partage des données

Pour commencer à interroger des données dans l’unité de partage des données, créez une base de données à partir d’une unité de partage des données. Vous ne pouvez créer qu’une seule base de données à partir d’une unité de partage des données spécifié.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Choisissez Datashares. La liste d’unités de partage des données s’affiche.
4. Dans la section Datashares from other clusters (Datashares provenant d'autres clusters), choisissez Connect to database (Connexion à la base de données). Pour plus d'informations, consultez Connexion à une base de données.
5. Choisissez une unité de partage des données à partir duquel vous voulez créer des bases de données, puis choisissez Créer une base de données à partir d’une unité de partage des données. La page Créer une base de données à partir d’une unité de partage des données s’affiche.
6. Dans Database name (Nom de base de données), spécifiez un nom de base de données. Le nom de base de données doit comporter 1 à 64 caractères alphanumériques (minuscules uniquement) et il ne peut pas s’agir d’un mot réservé.
7. Choisissez Créer.

Une fois la base de données créée, vous pouvez interroger les données de la base de données.

Gestion des partages AWS Data Exchange de données

Avec Amazon Redshift, vous pouvez partager et recevoir des données en direct en toute sécurité AWS Data Exchange sans avoir à créer et à gérer des extraits de données ou des pipelines. La gestion des AWS Data Exchange partages de données vous permet de vous abonner à des produits de données tiers et d'intégrer des flux de données en direct directement dans votre entrepôt de données Amazon Redshift. Les sections suivantes présentent la gestion des partages AWS Data Exchange de données au sein de vos clusters Amazon Redshift.

Création d'ensembles de données sur AWS Data Exchange

Créez des ensembles de données sur AWS Data Exchange.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Choisissez Datashares.
4. Dans la section Partages de données créés dans mon compte, choisissez un AWS Data Exchange partage de données.
5. Choisissez Créer un ensemble de données sur AWS Data Exchange. Pour plus d'informations, consultez Publication d'un nouveau produit.

Modification de partages AWS Data Exchange de données

Modifiez les AWS Data Exchange partages de données à l'aide de la console. Connectez-vous d'abord à une base de données pour afficher la liste des unités de partage des données créés dans votre compte.

Pour les AWS Data Exchange partages de données, vous ne pouvez pas apporter de modifications aux consommateurs de données.

Pour modifier le paramètre accessible au public pour les AWS Data Exchange partages de données, utilisez l'éditeur de requête v2. Amazon Redshift génère une valeur ponctuelle aléatoire pour définir la variable de séance afin d'autoriser la désactivation de ce paramètre. Pour de plus amples informations, veuillez consulter Notes d'utilisation d'ALTER DATASHARE.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Dans le menu du navigateur, choisissez Editor (Éditeur), puis Query editor v2 (Éditeur de requête v2).
4. S'il s'agit de la première fois que vous utilisez l'éditeur de requêtes v2, configurez votre Compte AWS. Par défaut, une clé AWS détenue est utilisée pour chiffrer les ressources. Pour plus d'informations sur la configuration de votre Compte AWS, consultez la section Configuration de votre Compte AWS dans le guide de gestion Amazon Redshift.

5. Pour vous connecter au cluster dans lequel se trouve votre AWS Data Exchange partage de données, choisissez Database et le nom du cluster dans le panneau d'arborescence. Si vous y êtes invité, saisissez les paramètres de connexion.
6. Copiez l'instruction SQL suivante. L'exemple suivant montre la modification du le paramètre accessible au public de l'unité de partage des données salesshare.

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
```

7. Pour exécuter l'instruction SQL copiée, choisissez Queries (Requêtes) et collez l'instruction SQL copiée dans la zone de requête. Choisissez ensuite Next (Suivant).

   Une erreur apparaît comme suit :

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
ERROR:  Alter of ADX-managed datashare salesshare requires session variable
 datashare_break_glass_session_var to be set to value 'c670ba4db22f4b'
```

   La valeur « c670ba4db22f4b » est une valeur ponctuelle aléatoire générée par Amazon Redshift lorsqu'une opération non recommandée se produit.
8. Copiez et collez l'exemple d'instruction suivant dans l'éditeur de requêtes. Ensuite, exécutez la commande. La commande `SET datashare_break_glass_session_var` applique une autorisation afin d'autoriser une opération non recommandée pour une unité de partage des données AWS Data Exchange .

```
SET datashare_break_glass_session_var to 'c670ba4db22f4b';
```

9. Exécutez à nouveau l'instruction ALTER DATASHARE.

```
ALTER DATASHARE salesshare;
```

Amazon Redshift met à jour votre unité de partage des données avec les modifications.

Supprimer les AWS Data Exchange partages de données créés dans votre compte

Supprimez les AWS Data Exchange partages de données créés dans votre compte à l'aide de la console. Connectez-vous d'abord à une base de données pour afficher la liste des unités de partage des données créés dans votre compte.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Dans le menu du navigateur, choisissez Editor (Éditeur), puis Query editor v2 (Éditeur de requête v2).
4. S'il s'agit de la première fois que vous utilisez l'éditeur de requêtes v2, configurez votre Compte AWS. Par défaut, une clé AWS détenue est utilisée pour chiffrer les ressources. Pour plus d'informations sur la configuration de votre Compte AWS, consultez la section Configuration de votre Compte AWS dans le guide de gestion Amazon Redshift.
5. Pour vous connecter au cluster dans lequel se trouve votre AWS Data Exchange partage de données, choisissez Database et le nom du cluster dans le panneau d'arborescence. Si vous y êtes invité, saisissez les paramètres de connexion.
6. Copiez l'instruction SQL suivante. L'exemple suivant montre la suppression de l'unité de partage des données salesshare.

```
DROP DATASHARE salesshare
```

7. Pour exécuter l'instruction SQL copiée, choisissez Queries (Requêtes) et collez l'instruction SQL copiée dans la zone de requête. Choisissez ensuite Next (Suivant).

   Une erreur apparaît comme suit :

```
ERROR:  Drop of ADX-managed datashare salesshare requires session variable
 datashare_break_glass_session_var to be set to value '620c871f890c49'
```

   La valeur « 620c871f890c49 » est une valeur ponctuelle aléatoire générée par Amazon Redshift lorsqu'une opération non recommandée se produit.

8. Copiez et collez l'exemple d'instruction suivant dans l'éditeur de requêtes. Ensuite, exécutez la commande. La commande `SET datashare_break_glass_session_var` applique une autorisation afin d'autoriser une opération non recommandée pour une unité de partage des données AWS Data Exchange .

```
SET datashare_break_glass_session_var to '620c871f890c49';
```

9. Exécutez à nouveau l'instruction DROP DATASHARE.

```
DROP DATASHARE salesshare;
```

Une fois les unités de partage des données supprimées, les consommateurs d'unités de partage des données perdent l'accès aux unités de partage des données.

La suppression d'un partage de AWS Data Exchange données partagé peut enfreindre les conditions du produit de données contenues dans. AWS Data Exchange

Commencer à partager des données en lecture seule avec l'interface SQL

Avec Amazon Redshift, vous pouvez partager des données en toute sécurité entre les clusters Amazon Redshift, ce qui permet aux consommateurs de données d'interroger et d'accéder à des données en temps réel sans les copier ni les répliquer. Le partage de données vous permet de créer et de configurer des partages de données, qui sont des objets côté producteur qui font référence aux objets de base de données que vous souhaitez partager.

Vous pouvez partager des données à des fins de lecture entre différents clusters Amazon Redshift sein d'un compte ou entre des Comptes AWS, ou entre des Régions AWS.

Rubriques

- Partage de l'accès en lecture aux données au sein d'un Compte AWS
- Utiliser les vues dans le partage de données Amazon Redshift
- Ajouter des tables de lac de données à un partage de données
- Partage de données entre Comptes AWS
- Partage de données entre Régions AWS
- Partage de données Amazon Redshift sous licence sur AWS Data Exchange
- Commencer à utiliser les AWS Lake Formation partages de données gérés

Partage de l'accès en lecture aux données au sein d'un Compte AWS

Avec Amazon Redshift, vous pouvez partager l'accès en lecture aux données entre différents utilisateurs ou groupes de bases de données au sein d'un même AWS compte. Cette fonctionnalité vous permet de contrôler les privilèges d'accès aux données à un niveau granulaire, en veillant à ce que seuls les utilisateurs ou les groupes autorisés puissent lire des ensembles de données spécifiques.

Partagez des données à des fins de lecture en tant que producteur, administrateur ou propriétaire de base de données

1. Créez des unités de partage des données dans votre cluster. Pour plus d'informations, consultez CREATE DATASHARE.

   ```
   CREATE DATASHARE salesshare;
   ```

   Les super-utilisateurs de cluster et les propriétaires de base de données peuvent créer des datashares. Chaque datashare est associé à une base de données lors de sa création. Seuls les objets de cette base de données peuvent être partagés dans ce datashare. Plusieurs datashares peuvent être créés sur la même base de données avec la même précision d'objets ou une précision différente. Il n'y a pas de limite au nombre d'unités de partage des données qu'un cluster peut créer.

   Vous pouvez également créer des datashares via la console Amazon Redshift. Pour plus d'informations, consultez Création d'un partage de données.

2. Déléguer des autorisations pour opérer sur l'unité de partage des données. Pour plus d'informations, consultez GRANT ou REVOKE.

   L'exemple suivant attribue des autorisations à `dbuser` sur `salesshare`.

   ```
   GRANT ALTER, SHARE ON DATASHARE salesshare TO dbuser;
   ```

   Les super-utilisateurs de cluster et les propriétaires de l'unité de partage des données peuvent accorder ou révoquer des autorisations de modification sur l'unité de partage des données à d'autres utilisateurs.

3. Ajoutez ou supprimez des objets des unités de partage des données. Pour ajouter des objets à une unité de partage des données, ajoutez le schéma avant d'ajouter des objets. Lorsque vous ajoutez un schéma, Amazon Redshift n'ajoute pas tous les objets qu'il contient. Assurez-vous de les ajouter explicitement. Pour plus d'informations, consultez ALTER DATASHARE.

   ```
   ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
   ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
   ```

   Vous pouvez également ajouter des vues à une unité de partage des données.

```
CREATE VIEW public.sales_data_summary_view AS SELECT * FROM
 public.tickit_sales_redshift;
ALTER DATASHARE salesshare ADD VIEW public.sales_data_summary_view;
```

Utilisez ALTER DATASHARE pour partager des schémas, des tables, des vues et des fonctions dans un schéma donné. Les super-utilisateurs, les propriétaires d'unité de partage des données ou les utilisateurs disposant des autorisations ALTER ou ALL sur l'unité de partage des données peuvent modifier l'unité de partage des données pour y ajouter des objets ou en supprimer. Les utilisateurs doivent avoir les autorisations nécessaires pour ajouter ou supprimer des objets de l'unité de partage des données. Les utilisateurs doivent également être les propriétaires des objets ou disposer des autorisations SELECT, USAGE ou ALL sur les objets.

Vous pouvez également utiliser GRANT pour ajouter des objets au partage de données. Cet exemple montre comment :

```
GRANT SELECT ON TABLE public.tickit_sales_redshift TO DATASHARE salesshare;
```

Cette syntaxe est fonctionnellement équivalente à. `ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;`

Utilisez la clause INCLUDENEW pour ajouter au partage de données les nouvelles tables, vues ou fonctions SQL définies par l'utilisateur (UDFs) créées dans un schéma spécifié. Seuls les super-utilisateurs peuvent modifier cette propriété pour chaque paire datashare-schéma.

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
ALTER DATASHARE salesshare SET INCLUDENEW = TRUE FOR SCHEMA PUBLIC;
```

Vous pouvez également utiliser la console Amazon Redshift pour ajouter ou supprimer des objets des unités de partage des données. Pour plus d'informations, consultez Ajouter des objets de partage de données aux partages de données, Supprimer des objets d'unité de partage des données des unités de partage des données et Modification des unités de partage des données créées dans votre compte.

4. Ajoutez ou supprimez des consommateurs des datashares. L'exemple suivant ajoute l'espace de noms du consommateur à`salesshare`. L'espace de noms est l'identifiant global unique d'espace de noms (GUID) du cluster consommateur sur le compte. Pour plus d'informations, consultez GRANT ou REVOKE.

```
GRANT USAGE ON DATASHARE salesshare TO NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Vous ne pouvez accorder des autorisations qu'à un seul consommateur d'unité de partage des données dans une instruction GRANT.

Les super-utilisateurs de cluster et les propriétaires d'objets d'unité de partage des données, ou les utilisateurs qui ont l'autorisation SHARE sur l'unité de partage des données, peuvent ajouter des consommateurs dans une unité de partage des données, ou en supprimer. Pour ce faire, ils utilisent GRANT USAGE ou REVOKE USAGE.

Pour rechercher l'espace de noms du cluster que vous voyez actuellement, vous pouvez utiliser la commande SELECT CURRENT_NAMESPACE. Pour trouver l'espace de noms d'un autre cluster au sein d'un même cluster Compte AWS, rendez-vous sur la page de détails du cluster de console Amazon Redshift. Sur cette page, recherchez le champ d'espace de noms nouvellement ajouté.

Vous pouvez également utiliser la console Amazon Redshift pour ajouter ou supprimer des consommateurs de données pour les unités de partage des données. Pour plus d'informations, consultez Ajouter des consommateurs de données aux partages de données et Supprimer des consommateurs de données des unités de partage des données.

5. (Facultatif) Ajoutez des restrictions de sécurité à l'unité de partage des données. L'exemple suivant montre que le cluster consommateur avec un accès IP public est autorisé à lire l'unité de partage des données. Pour plus d'informations, consultez ALTER DATASHARE.

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE = TRUE;
```

Vous pouvez modifier les propriétés du type de consommateurs après la création de l'unité de partage des données. Par exemple, vous pouvez définir que les clusters qui souhaitent consommer des données d'une unité de partage des données donné ne peuvent pas être accessibles publiquement. Les requêtes provenant de clusters consommateur qui ne respectent pas les restrictions de sécurité spécifiées dans l'unité de partage des données sont rejetées au moment de l'exécution de la requête.

Vous pouvez également utiliser la console Amazon Redshift pour modifier les datashares. Pour plus d'informations, consultez Modification des unités de partage des données créées dans votre compte.

6. Répertorie les unités de partage des données créées dans le cluster et examine le contenu de l'unité de partage des données.

   L'exemple suivant montre comment afficher les informations d'une unité de partage des données nommée `salesshare`.

```
DESC DATASHARE salesshare;

 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name          |   include_new
-------------------+--------------------------------------+------------
+------------+-------------+--------------------------------+------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_users_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_venue_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_category_redshift|
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_date_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_event_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_listing_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_sales_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | schema      | public                         |  t
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | view        | public.sales_data_summary_view |
```

   L'exemple suivant montre comment afficher les datashares sortants dans un cluster producteur.

```
SHOW DATASHARES LIKE 'sales%';
```

   La sortie ressemble à ce qui suit.

```
share_name | share_owner  | source_database | consumer_database | share_type |
 createdate      | is_publicaccessible  | share_acl | producer_account |
 producer_namespace
```

```
----------+-------------+----------------+------------------+------------
+--------------------+---------------------+----------+-----------------
+--------------------------------------
salesshare |    100       | dev             |                  |  OUTBOUND
 | 2020-12-09 02:27:08 |          True        |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

Pour plus d'informations, consultez DESC DATASHARE et SHOW DATASHARES.

Vous pouvez également utiliser SVV_DATASHARES, SVV_DATASHARE_CONSUMERS, et SVV_DATASHARE_OBJECTS pour afficher les datashares, les objets du datashare et les consommateurs de datashare.

7. Supprimer des unités de partage des données. Pour plus d'informations, consultez DROP DATASHARE.

   Vous pouvez supprimer les objets d'unité de partage des données à tout moment à l'aide de DROP DATASHARE. Les super-utilisateurs de cluster et les propriétaires d'unité de partage des données peuvent supprimer des unités de partage des données.

   L'exemple suivant supprime une unité de partage des données appelé `salesshare`.

   ```
   DROP DATASHARE salesshare;
   ```

   Vous pouvez également utiliser la console Amazon Redshift pour supprimer des unités de partage des données. Pour plus d'informations, consultez Supprimer un partage de données créé dans votre compte.

8. Utilisez ALTER DATASHARE pour supprimer des objets des datashares à n'importe quel point du datashare. Utilisez REVOKE USAGE ON pour révoquer les autorisations de certains consommateurs sur l'unité de partage des données. Il révoque les autorisations USAGE sur les objets dans une unité de partage des données et arrête instantanément l'accès à tous les clusters consommateur. Répertorier des unités de partage des données et des requêtes de métadonnées, telles que la liste des bases de données et des tables, ne renvoie pas les objets partagés après la révocation de l'accès.

   ```
   ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
   ```

Vous pouvez également utiliser la console Amazon Redshift pour modifier les datashares. Pour plus d'informations, consultez Modification des unités de partage des données créées dans votre compte.

9. Révoquez l'accès à l'unité de partage des données depuis les espaces de noms si vous ne souhaitez plus partager les données avec les consommateurs.

```
REVOKE USAGE ON DATASHARE salesshare FROM NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Vous pouvez également utiliser la console Amazon Redshift pour modifier les datashares. Pour de plus amples informations, veuillez consulter Modification des unités de partage des données créées dans votre compte.

Partagez des données à des fins de lecture en tant qu'administrateur de consommateurs

1. Répertorier les unités de partage des données mises à votre disposition et afficher leur contenu. Pour plus d'informations, consultez DESC DATASHARE et SHOW DATASHARES.

   L'exemple suivant montre comment afficher les informations des unités de partage des données entrants d'un espace de noms producteur spécifié. Lorsque vous exécutez DESC DATASHARE en tant qu'administrateur client, vous devez spécifier l'option NAMESPACE pour afficher les partages de données entrants.

```
DESC DATASHARE salesshare OF NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';


 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name           |   include_new
-------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_users_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_venue_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_category_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_date_redshift     |
```

```
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_event_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_listing_redshift  |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_sales_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | schema      | public                          |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | view        | public.sales_data_summary_view  |
```

Seuls les super-utilisateurs de cluster peuvent le faire. Vous pouvez également utiliser SVV_DATASHARES pour afficher les unités de partage des données et SVV_DATASHARE_OBJECTS pour afficher les objets dans l'unité de partage des données.

L'exemple suivant affiche les datashares entrants dans un cluster consommateur.

```
SHOW DATASHARES LIKE 'sales%';


 share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
------------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+--------------------------------------
 salesshare |             |                 |                   | INBOUND
    |            |         t           |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

2. En tant que super-utilisateur de base de données, vous pouvez créer des bases de données locales qui font référence aux unités de partage des données. Pour de plus amples informations, veuillez consulter CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Si vous souhaitez un contrôle plus précis de l'accès aux objets de la base de données locale, utilisez la clause WITH PERMISSIONS quand vous créez la base de données. Cela vous permet d'accorder des autorisations de niveau objet pour les objets de la base de données à l'étape 4.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Vous pouvez voir les bases de données que vous avez créées à partir de l'unité de partage des données en interrogeant la vue SVV_REDSHIFT_DATABASES. Vous pouvez vous connecter directement à ces bases de données ou vous connecter à une base de données locale sur votre cluster client et effectuer une requête entre bases de données pour interroger les données des bases de données partagées. Vous ne pouvez pas créer une unité de partage des données au-dessus des objets de base de données créés à partir d'une unité de partage des données existant. Toutefois, vous pouvez copier les données dans une table distincte du cluster consommateur, effectuer le traitement nécessaire, puis partager les nouveaux objets créés.

Vous pouvez également utiliser la console Amazon Redshift pour créer des bases de données à partir des unités de partage des données. Pour plus d'informations, consultez Créer des bases de données à partir d'unités de partage des données.

3. (Facultatif) Créez des schémas externes pour faire référence et attribuer des autorisations détaillées à des schémas spécifiques dans la base de données consommateur importée sur le cluster consommateur. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL SCHEMA.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA
 'public';
```

4. Selon les besoins, accordez aux utilisateurs et rôles du cluster consommateur des autorisations sur les bases de données et références de schéma créées à partir des unités de partage des données. Pour plus d'informations, consultez GRANT ou REVOKE.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Si vous avez créé votre base de données sans la clause WITH PERMISSIONS, vous ne pouvez attribuer des autorisations sur l'ensemble de la base de données créée à partir de l'unité de partage des données qu'à vos utilisateurs et groupes. Dans certains cas, vous avez besoin de contrôles précis sur un sous-ensemble d'objets de base de données créés à partir de l'unité de partage des données. Si tel est le cas, vous pouvez créer une référence de schéma externe

tournée vers des schémas spécifiques dans l'unité de partage des données (comme décrit à l'étape précédente) et fournir des autorisations détaillées au niveau du schéma.

Vous pouvez également créer des vues à liaison tardive sur les objets partagés et les utiliser pour attribuer des autorisations détaillées. Vous pouvez également envisager que les clusters producteur créent des unités de partage des données supplémentaires pour vous avec la précision requise.

Si vous avez créé votre base de données avec la clause WITH PERMISSIONS à l'étape 2, vous devez attribuer des autorisations de niveau objet dans la base de données partagée. Un utilisateur disposant uniquement de l'autorisation USAGE ne peut accéder à aucun objet d'une base de données créée avec la clause WITH PERMISSIONS tant qu'il n'a pas obtenu d'autorisations de niveau objet supplémentaires.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

5. Interroger les données dans les objets partagés dans les unités de partage des données.

   Les utilisateurs et les rôles disposant d'autorisations sur les bases de données consommateur et les schémas sur les clusters consommateur peuvent explorer et naviguer entre les métadonnées de tous les objets partagés. Ils peuvent également explorer et naviguer entre les objets locaux dans un cluster consommateur. Pour ce faire, ils utilisent des pilotes JDBC ou ODBC ou des vues SVV_ALL et SVV_REDSHIFT.

   Les clusters producteur peuvent avoir de nombreux schémas dans la base de données, les tables et les vues au sein de chaque schéma. Les utilisateurs du côté consommateur ne peuvent voir que le sous-ensemble d'objets mis à disposition via l'unité de partage des données. Ces utilisateurs ne peuvent pas voir l'intégralité des métadonnées du cluster producteur. Cette approche permet de fournir un contrôle détaillé de la sécurité des métadonnées avec le partage des données.

   Vous continuez à vous connecter aux bases de données de cluster locales. Mais maintenant, vous pouvez également lire à partir des bases de données et des schémas qui sont créés à partir de l'unité de partage des données en utilisant la notation database.schema.table en trois parties. Vous pouvez effectuer des requêtes qui s'étendent sur toutes les bases de données qui sont visibles pour vous. Il peut s'agir de bases de données locales sur le cluster ou de bases de données créées à partir des unités de partage des données. Les clusters consommateur ne peuvent pas se connecter aux bases de données créées à partir des unités de partage des données.

Vous pouvez accéder aux données à l'aide d'une qualification complète. Pour plus d'informations, consultez Exemples de requêtes entre bases de données.

```
SELECT * FROM sales_db.public.tickit_sales_redshift ORDER BY 1,2 LIMIT 5;

 salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime
---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------
       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |    728.00 |
   109.20 | 2008-02-18 02:36:48
       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |     76.00 |
    11.40 | 2008-06-06 05:00:16
       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |    350.00 |
    52.50 | 2008-06-06 08:26:17
       4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |    175.00 |
    26.25 | 2008-06-09 08:38:52
       5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |    154.00 |
    23.10 | 2008-08-31 09:17:02
```

Vous ne pouvez utiliser les instructions SELECT que sur les objets partagés. Toutefois, vous pouvez créer des tables dans le cluster consommateur en interrogeant les données des objets partagés dans une base de données locale différente.

En plus des requêtes, les consommateurs peuvent créer des vues sur des objets partagés. Seules les vues à liaison tardive ou les vues matérialisées sont prises en charge. Amazon Redshift ne prend pas en charge les vues standard sur les données partagées. Les vues créées par les consommateurs peuvent s'étendre sur plusieurs bases de données locales ou bases de données créées à partir d'unités de partage des données. Pour de plus amples informations, veuillez consulter CREATE VIEW.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.
CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Utiliser les vues dans le partage de données Amazon Redshift

Un cluster producteur peut partager des vues standard, à liaison tardive et matérialisées. Lorsque vous partagez des vues standard ou à liaison tardive, vous n'avez pas besoin de partager les tables de base. Le tableau suivant montre comment les vues sont prises en charge avec le partage de données.

Nom de la vue	Est-ce que cette vue peut être ajoutée à une unité de partage des données ?	Un consommateur peut-il créer cette vue sur des objets d'unité de partage des données entre des clusters ?
Vue standard	Oui	Non
Vues à liaison tardive	Oui	Oui
Vues matérialisées	Oui	Oui, mais uniquement avec une actualisation complète

La requête suivante affiche la sortie d'une vue standard prise en charge par le partage de données. Pour plus d'informations sur la définition des vues régulières, consultez CREATE VIEW.

```
SELECT * FROM tickit_db.public.myevent_regular_vw
ORDER BY eventid LIMIT 5;

   eventid  |  eventname
  ----------+-------------
     3835   | LeAnn Rimes
     3967   | LeAnn Rimes
     4856   | LeAnn Rimes
     4948   | LeAnn Rimes
     5131   | LeAnn Rimes
```

La requête suivante affiche la sortie d'une vue à liaison tardive prise en charge par le partage de données. Pour plus d'informations sur la définition des vues à liaison tardive, consultez CREATE VIEW.

```
SELECT * FROM tickit_db.public.event_lbv
ORDER BY eventid LIMIT 5;

 eventid | venueid | catid | dateid |          eventname           |      starttime
 --------+---------+-------+--------+------------------------------
+---------------------
     1   |   305   |   8   |  1851  |        Gotterdammerung       | 2008-01-25
 14:30:00
     2   |   306   |   8   |  2114  |           Boris Godunov      | 2008-10-15
 20:00:00
     3   |   302   |   8   |  1935  |              Salome          | 2008-04-19
 14:30:00
     4   |   309   |   8   |  2090  |  La Cenerentola (Cinderella) | 2008-09-21
 14:30:00
     5   |   302   |   8   |  1982  |          Il Trovatore        | 2008-06-05
 19:00:00
```

La requête suivante affiche la sortie d'une vue matérialisée prise en charge par le partage de données. Pour plus d'informations sur la définition des vues matérialisées, consultez CREATE MATERIALIZED VIEW.

```
SELECT * FROM tickit_db.public.tickets_mv;

  catgroup | qtysold
 ----------+---------
  Concerts | 195444
    Shows  | 149905
```

Vous pouvez gérer des tables communes à tous les locataires d'un cluster producteur. Vous pouvez également partager des sous-ensembles de données filtrées par colonnes de dimension, tels que `tenant_id` (`account_id` ou `namespace_id`), aux clusters consommateur. Pour ce faire, vous pouvez définir une vue sur la table de base avec un filtre sur ces colonnes d'ID, par exemple `current_aws_account = tenant_id`. Côté consommateur, lorsque vous interrogez la vue, vous ne voyez que les lignes correspondant à votre compte. Pour ce faire, vous pouvez utiliser les fonctions contextuelles Amazon Redshift `current_aws_account` et `current_namespace`.

La requête suivante renvoie l'ID de compte dans lequel réside le cluster Amazon Redshift actuel. Vous pouvez exécuter cette requête si vous êtes connecté à Amazon Redshift.

```
select current_user, current_aws_account;
```

```
current_user | current_aws_account
-------------+--------------------
dwuser       |    111111111111
(1row)
```

La requête suivante renvoie l'espace de noms du cluster Amazon Redshift actuel. Vous pouvez exécuter cette requête si vous êtes connecté à la base de données.

```
select current_user, current_namespace;

current_user | current_namespace
-------------+--------------------------------------
dwuser       | 86b5169f-01dc-4a6f-9fbb-e2e24359e9a8
(1 row)
```

Actualisation incrémentielle des vues matérialisées dans un partage de données

Amazon Redshift prend en charge l'actualisation incrémentielle des vues matérialisées dans un partage de données client lorsque les tables de base sont partagées. L'actualisation incrémentielle est une opération au cours de laquelle Amazon Redshift identifie les modifications apportées à la table de base ou aux tables après l'actualisation précédente et met à jour uniquement les enregistrements correspondants dans la vue matérialisée. Pour plus d'informations sur ce comportement, consultez CREATE MATERIALIZED VIEW.

Ajouter des tables de lac de données à un partage de données

Grâce à un partage de données, un producteur de données peut partager en toute sécurité des objets de base de données de granularité fine, tels que des schémas et des tables, avec des consommateurs appartenant au même AWS compte ou à des comptes différents. Le producteur peut également partager des objets entre les régions. Cette rubrique décrit comment ajouter des objets d'un lac de données, en particulier du catalogue de AWS Glue données, à un partage de données. Il couvre deux cas d'utilisation :

- Ajouter une vue de liaison tardive à un partage de données qui fait référence à une table provenant d'un lac de données — C'est pratique pour le consommateur, car la configuration préliminaire, telle que la définition des autorisations sur les données sources externes, par exemple avec Lake Formation, est probablement déjà terminée. Un autre avantage est qu'une vue ajoutée à un partage de données permet de joindre des tables issues du lac de données à des tables natives Redshift.

- Ajout direct d'une table à partir d'un schéma externe à un partage de données : les objets du lac de données sont ainsi mis à la disposition des consommateurs sans couches ni logique supplémentaires. Les consommateurs peuvent interroger la table ou la joindre aux tables du consommateur.

Ces cas s'appliquent une fois que vous avez référencé une table du catalogue de AWS données dans Redshift à l'aide de CREATE EXTERNAL SCHEMA. N'importe quelle table du catalogue de AWS données peut être la source.

> **Note**
>
> Les tables de lac de données que vous ajoutez à un partage de données peuvent inclure des tables enregistrées auprès de Lake Formation et des tables de catalogue de AWS Glue données.

Création d'un schéma externe et d'une table externe

Vous créez un schéma externe et une table externe afin de les ajouter au partage de données dans les sections qui suivent. Il s'agit d'étapes préliminaires. Si vous l'avez déjà fait, vous pouvez ignorer cette section.

1. Sur le producteur, créez un schéma externe qui fait référence aux données du lac de données stockées dans Amazon S3. Le schéma externe fait référence au AWS Glue Data Catalog. Le rôle et la région de l'échantillon en sont des exemples :

```
CREATE EXTERNAL SCHEMA external_schema_name FROM DATA CATALOG
DATABASE 'glue_database_name'
IAM_ROLE 'arn:aws:iam::123456789012:role/sample-role'
REGION 'us-east-1';
```

2. Créez une table de lac de données dans le schéma externe.

```
CREATE EXTERNAL TABLE external_schema_name.sales(
salesid INTEGER,
sellerid INTEGER,
buyerid INTEGER,
saledate DATE,
pricepaid DECIMAL(8,2))
ROW FORMAT delimited
```

```
FIELDS TERMINATED BY '\t'
STORED AS textfile
LOCATION 's3://redshift-downloads/tickit/spectrum/sales/';
```

L'échantillon inclut le`LOCATION`. Il doit être dans le formulaire `s3://{bucket_name}/{folder}/` où le dossier est spécifié. Le dossier doit comporter au moins un caractère. Vous pouvez éventuellement inclure des sous-dossiers. Pour voir d'autres exemples de création de tables dans un lac de données, consultez la section Exemples de CREATE EXTERNAL TABLE.

> **Note**
>
> Le partage n'est pris en charge que pour les tables où le rôle IAM du producteur dispose d'un accès SELECT sur la table.

Ajoutez une vue de liaison tardive qui fait référence à une table de lac de données à un partage de données

Lorsque vous créez des tables basées sur un schéma externe issu du catalogue de AWS données et que vous souhaitez les ajouter à un partage de données, la méthode la plus courante consiste à ajouter une vue de liaison tardive Redshift faisant référence à la table que vous avez créée, qui contient les données du lac de données. La procédure suivante indique les étapes à suivre :

1. Créez une vue de liaison tardive qui fait référence à la table externe que vous avez créée précédemment :

   ```
   CREATE VIEW lbv AS
   select * from external_schema_name.sales, other_schema.t1
   WITH NO SCHEMA BINDING;
   ```

2. Ajoutez le schéma de vue au partage de données. Il s'agit du schéma local qui contient la vue de liaison tardive.

   ```
   ALTER DATASHARE dsx_datashare ADD SCHEMA public;
   ```

3. Ajoutez au partage de données le schéma qui contient la table référencée par la vue de liaison tardive. L'ajout du schéma est obligatoire pour toutes les tables de base référencées dans une vue ajoutée à un partage de données, que le schéma contienne des objets de base de données

locaux ou des objets provenant d'un lac de données. Notez que vous devez ajouter ce schéma avant d'ajouter la vue de liaison tardive.

```
ALTER DATASHARE dsx_datashare ADD SCHEMA external_schema_name;
ALTER DATASHARE dsx_datashare ADD SCHEMA other_schema;
```

4. Ajoutez la vue au partage de données à l'aide d'une commande SQL. Notez que le nom de la table inclut le préfixe du schéma.

```
ALTER DATASHARE my_datashare ADD TABLE public.lbv;
```

5. Vérifiez que la vue et les schémas ont bien été ajoutés au partage de données :

```
SELECT * FROM svv_datashare_objects WHERE share_name = 'my_datashare';
```

6. L'administrateur du consommateur crée une base de données à partir du partage de données, puis en autorise l'utilisation aux utilisateurs consommateurs.

Une fois les étapes terminées, les utilisateurs consommateurs de la base de données ayant accès à la vue de partage de données peuvent interroger les données.

Ajouter une table de lac de données directement dans un partage de données

L'ajout d'une table dans un schéma externe à un partage de données est similaire à l'ajout d'une vue. Cela fonctionne bien dans le cas où un consommateur souhaite interroger la table du lac de données dans son état d'origine, ou s'il souhaite la joindre aux tables de l'entrepôt de données des consommateurs. Les étapes suivantes vous montrent comment ajouter une table de lac de données à un partage de données à l'aide de SQL.

1. Créez un schéma externe et une table externe, comme décrit dans la première section de cette rubrique.
2. Découvrez les tables existantes dans le schéma externe pour vérifier que la table que vous avez créée est disponible :

```
SELECT * FROM svv_external_tables WHERE schemaname = 'external_schema_name';
```

3. Ajoutez le schéma externe au partage de données :

```
ALTER DATASHARE my_datashare ADD SCHEMA external_schema_name;
```

4. Ajoutez la table externe au partage de données. Notez que le nom de la table inclut le préfixe du schéma :

   ```
   ALTER DATASHARE my_datashare ADD TABLE external_schema_name.sales;
   ```

5. Vérifiez que la table a bien été ajoutée au partage de données :

   ```
   SELECT * FROM svv_datashare_objects WHERE share_name = 'my_datashare';
   ```

   Pour des instructions plus détaillées, voir Partage de l'accès en lecture aux données d'un AWS compte.

6. Sur le consommateur, qui est la base de données recevant les données partagées, l'administrateur associe le partage de données pour que les utilisateurs puissent les interroger sur les tables partagées. Pour plus d'informations sur la manière d'effectuer cette étape, consultez la section Gestion des partages de données provenant d'autres comptes en tant que consommateur.

Une fois que les administrateurs ont terminé les étapes, les utilisateurs de la base de données du consommateur peuvent écrire des requêtes pour récupérer les données de la table partagée et les joindre aux autres tables du consommateur.

Remarques d'utilisation relatives à l'ajout d'objets de lac de données à un partage de données

Lorsque vous utilisez des tables et des vues issues d'un lac de données dans un partage de données, vous devez tenir compte de plusieurs points :

- Connexion avec AWS CloudTrail : le compte du producteur de données peut utiliser AWS CloudTrail des journaux pour vérifier l'accès aux tables de lacs de données partagées via un partage de données :
  - Utilisation des données des journaux pour contrôler l'accès aux données : les CloudTrail journaux enregistrent des informations sur les personnes qui accèdent aux tables partagées, y compris les producteurs et les consommateurs de partages de données Redshift. Les identifiants sont disponibles dans le `ExternalId` champ situé sous les `AssumeRole` CloudTrail journaux. Le propriétaire des données peut configurer des limites supplémentaires d'accès aux données

dans une politique IAM au moyen d'actions. Pour plus d'informations sur la définition de l'accès aux données par le biais de politiques, voir Accès aux AWS comptes détenus par des tiers.

- Sécurité et autorisations des consommateurs : pour les tables enregistrées par Lake Formation, les ressources Amazon S3 sont sécurisées par Lake Formation et mises à disposition à l'aide des informations d'identification fournies par Lake Formation.

Considérations relatives à la facturation pour l'ajout d'objets de lac de données à un partage de données

Le tableau suivant détaille la manière dont les coûts sont attribués pour le stockage et la numérisation d'objets de lac de données dans un partage de données :

- Lorsqu'un consommateur demande des objets partagés à partir d'un lac de données, le coût de numérisation est facturé au consommateur.
  - Lorsque le consommateur est un cluster provisionné, Redshift utilise Redshift Spectrum pour analyser les données Amazon S3. Par conséquent, le coût du spectre est facturé au compte du consommateur.
  - Lorsque le consommateur est un groupe de travail Amazon Redshift Serverless, Spectrum n'est soumis à aucun frais distinct.
- Les frais de stockage et d'exploitation d'Amazon S3, tels que la mise en vente de compartiments, sont facturés au compte propriétaire de chaque compartiment Amazon S3.

Pour plus d'informations concernant la facturation d'Amazon Redshift Serverless, consultez la section Facturation d'Amazon Redshift Serverless. Des informations supplémentaires sur la facturation et les prix sont disponibles sur la page de tarification d'Amazon Redshift.

Partage de données entre Comptes AWS

Vous pouvez partager des données à des fins de lecture entre des Comptes AWS. Le partage de données entre des Comptes AWS fonctionne de la même manière que le partage de données au sein d’un compte. La différence est qu’une négociation bidirectionnelle est requise pour partager des données entre des Comptes AWS. Un administrateur de compte producteur peut autoriser des comptes consommateurs à accéder à des unités de partage des données ou choisir de ne pas autoriser l’accès. Pour utiliser une unité de partage des données autorisée, un administrateur de compte consommateur peut associer l’unité de partage des données. L'administrateur peut associer le partage de données à un ensemble Compte AWS ou à des clusters spécifiques du compte client,

ou refuser le partage de données. Pour plus d'informations sur le partage de données au sein d'un compte, consultez Partage de l'accès en lecture aux données au sein d'un Compte AWS.

Un partage de données peut avoir des consommateurs de données qui sont soit des espaces de noms appartenant au même compte, soit différents. Comptes AWS Vous n'avez pas besoin de créer des unités de partage des données distincts pour le partage au sein d'un compte et le partage entre comptes.

Pour le partage de données entre comptes, les clusters producteur et consommateur doivent être chiffrés.

Lorsqu'ils partagent des données avec Comptes AWS, les administrateurs du producteur les partagent Compte AWS en tant qu'entité. Un administrateur client peut décider quels espaces de noms du compte client ont accès à un partage de données.

Rubriques

- actions du producteur et de l'administrateur
- Actions de l'administrateur de comptes consommateur
- actions de l'administrateur du consommateur

actions du producteur et de l'administrateur

Avec Amazon Redshift, vous pouvez effectuer des tâches administratives sur les clusters de producteurs afin de gérer l'ingestion de données et le traitement des charges.

Si vous êtes un producteur, un administrateur ou un propriétaire de base de données, procédez comme suit :

1. Créez des unités de partage des données dans votre cluster et ajoutez-y des objets d'unité de partage des données. Pour obtenir des étapes plus détaillées sur la création des unités de partage des données et l'ajout d'objets d'unité de partage des données à ces derniers, consultez Partage de l'accès en lecture aux données au sein d'un Compte AWS. Pour plus d'informations sur CREATE DATASHARE et ALTER DATASHARE, consultez CREATE DATASHARE et ALTER DATASHARE.

   L'exemple suivant ajoute différents objets d'unité de partage des données à l'unité de partage des données `salesshare`.

   ```
   -- Add schema to datashare
   ```

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;

-- Add table under schema to datashare
ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;

-- Add view to datashare
ALTER DATASHARE salesshare ADD TABLE public.sales_data_summary_view;

-- Add all existing tables and views under schema to datashare (does not include
 future table)
ALTER DATASHARE salesshare ADD ALL TABLES in schema public;
```

Vous pouvez également utiliser la console Amazon Redshift pour créer ou modifier des unités de partage des données. Pour plus d'informations, consultez Création d'un partage de données et Modification des unités de partage des données créées dans votre compte.

2. Déléguer des autorisations pour opérer sur l'unité de partage des données. Pour plus d'informations, consultez GRANT ou REVOKE.

   L'exemple suivant attribue des autorisations à `dbuser` sur `salesshare`.

   ```
   GRANT ALTER, SHARE ON DATASHARE salesshare TO dbuser;
   ```

   Les super-utilisateurs de cluster et les propriétaires de l'unité de partage des données peuvent accorder ou révoquer des autorisations de modification sur l'unité de partage des données à d'autres utilisateurs.

3. Ajoutez ou supprimez des consommateurs des unités de partage des données. L'exemple suivant ajoute l' Compte AWS ID à`salesshare`. Pour plus d'informations, consultez GRANT ou REVOKE.

   ```
   GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '123456789012';
   ```

   Vous ne pouvez accorder des autorisations qu'à un seul consommateur de données dans une instruction GRANT.

   Les super-utilisateurs de cluster et les propriétaires d'objets d'unité de partage des données ou les utilisateurs qui ont l'autorisation SHARE sur l'unité de partage des données peuvent ajouter ou supprimer des consommateurs d'une unité de partage des données. Pour ce faire, ils utilisent GRANT USAGE ou REVOKE USAGE.

Vous pouvez également utiliser la console Amazon Redshift pour ajouter ou supprimer des consommateurs de données pour les unités de partage des données. Pour plus d'informations, consultez Ajouter des consommateurs de données aux partages de données et Supprimer des consommateurs de données des unités de partage des données.

4. (Facultatif) Révoquez l'accès au partage de données Comptes AWS si vous ne souhaitez plus partager les données avec les consommateurs.

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '123456789012';
```

Si vous êtes administrateur de compte producteur, procédez comme suit :

Après avoir autorisé l'utilisation du Compte AWS, le statut du partage de données est. `pending_authorization` L'administrateur du compte producteur doit autoriser les unités de partage des données à l'aide de la console Amazon Redshift et choisir les consommateurs de données.

Connectez-vous à la https://console.aws.amazon.com/redshiftv2/. Choisissez ensuite les consommateurs de données auxquels autoriser l'accès à des unités de partage des données ou retirer l'autorisation. Les consommateurs de données autorisés reçoivent des notifications pour prendre des mesures sur les unités de partage des données. Si vous ajoutez un espace de noms en tant que consommateur de données, vous n'avez pas à effectuer d'autorisation. Une fois que les consommateurs de données sont autorisés, ils peuvent accéder aux objets d'unité de partage des données et créer une base de données consommateur pour interroger les données. Pour de plus amples informations, veuillez consulter Autoriser ou supprimer l'autorisation des unités de partage des données.

Partage des autorisations d'écriture sur les données entre les comptes

Avec Amazon Redshift, vous pouvez partager des données entre AWS comptes et accorder des autorisations d'écriture, ce qui permet la collaboration et le partage de données entre des équipes ou des organisations. Le partage de données entre comptes vous permet de créer un compte de fournisseur de données qui crée et gère des bases de données, des schémas et des tables, qui peuvent ensuite être partagés en toute sécurité avec des comptes de consommateurs de données. Les sections suivantes présentent le processus de configuration du partage de données entre comptes et d'octroi d'un accès en écriture dans Amazon Redshift.

Actions de l’administrateur de comptes consommateur

Avec Amazon Redshift, vous pouvez gérer les comptes des clients et contrôler leur accès à vos ressources d'entreposage de données.

Si vous êtes un administrateur de compte consommateur, procédez comme suit :

Pour associer un ou plusieurs partages de données partagés depuis d'autres comptes à l'intégralité Compte AWS ou à des espaces de noms spécifiques de votre compte, utilisez la console Amazon Redshift.

Connectez-vous à la https://console.aws.amazon.com/redshiftv2/. Associez ensuite un ou plusieurs partages de données partagés depuis d'autres comptes à l'ensemble Compte AWS ou à des espaces de noms spécifiques de votre compte. Pour de plus amples informations, veuillez consulter Associer un partage de données à un autre Compte AWS dans Amazon Redshift.

Une fois que les espaces de noms Compte AWS ou des espaces de noms spécifiques sont associés, les partages de données peuvent être utilisés. Vous pouvez également modifier l’association d’unité de partage des données à tout moment. Lorsque vous remplacez l'association d'espaces de noms individuels par un Compte AWS, Amazon Redshift remplace les espaces de noms par les informations. Compte AWS Lorsque l'association passe d'un espace de noms Compte AWS à un espace de noms spécifique, Amazon Redshift remplace les informations par Compte AWS les informations de l'espace de noms. Tous les espaces de noms du compte ont accès aux données.

actions de l'administrateur du consommateur

Avec Amazon Redshift, vous pouvez effectuer des tâches administratives sur des clusters de consommateurs afin de gérer l'ingestion de données et le traitement du chargement.

Si vous êtes un administrateur de consommateurs, procédez comme suit :

1. Répertoriez les unités de partage des données mises à votre disposition et affichez leur contenu. Le contenu des partages de données n'est disponible que lorsque l'administrateur du producteur a autorisé les partages de données et que l'administrateur du consommateur a accepté et associé les partages de données. Pour plus d’informations, consultez DESC DATASHARE et SHOW DATASHARES.

   L’exemple suivant montre comment afficher les informations des unités de partage des données entrants d’un espace de noms producteur spécifié. Lorsque vous exécutez DESC DATAHSARE en tant qu'administrateur client, vous devez spécifier le NAMESPACE et l'ID de compte pour afficher

les partages de données entrants. Pour les unités de partage des données sortantes, spécifiez le nom de l'unité de partage des données.

```
SHOW DATASHARES LIKE 'sales%';

share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
-----------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+--------------------------------------
salesshare |             |                 |                   | INBOUND    |
            |        t            |           | 123456789012    | 'dd8772e1-
d792-4fa4-996b-1870577efc0d'
```

```
DESC DATASHARE salesshare OF ACCOUNT '123456789012' NAMESPACE 'dd8772e1-
d792-4fa4-996b-1870577efc0d';


 producer_account |          producer_namespace          | share_type | share_name |
 object_type |           object_name
-----------------+--------------------------------------+------------+------------
+-------------+---------------------------------
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_users_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_venue_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_category_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_date_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_event_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_listing_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 table       | public.tickit_sales_redshift
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d | INBOUND    | salesshare |
 schema      | public
(8 rows)
```

Seuls les super-utilisateurs de cluster peuvent le faire. Vous pouvez également utiliser SVV_DATASHARES pour afficher les unités de partage des données et SVV_DATASHARE_OBJECTS pour afficher les objets dans l'unité de partage des données.

L'exemple suivant affiche les datashares entrants dans un cluster consommateur.

```
SELECT * FROM SVV_DATASHARES WHERE share_name LIKE 'sales%';

share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
-----------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+--------------------------------------
salesshare |             |                 |                   | INBOUND    |
            |        t            |           | 123456789012      | 'dd8772e1-
d792-4fa4-996b-1870577efc0d'
```

```
SELECT * FROM SVV_DATASHARE_OBJECTS WHERE share_name LIKE 'sales%';
 share_type | share_name | object_type |           object_name           |
 producer_account |          producer_namespace
------------+------------+-------------+---------------------------------
+------------------+--------------------------------------
 INBOUND    | salesshare | table       | public.tickit_users_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_venue_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_category_redshift |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_date_redshift     |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_event_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_listing_redshift  |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | table       | public.tickit_sales_redshift    |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
 INBOUND    | salesshare | schema      | public                          |
 123456789012     | dd8772e1-d792-4fa4-996b-1870577efc0d
(8 rows)
```

2. Créez des bases de données locales qui font référence aux unités de partage des données. Spécifiez la valeur NAMESPACE et l'ID de compte lors de la création de la base de données à partir de l'unité de partage des données. Pour de plus amples informations, veuillez consulter CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF ACCOUNT '123456789012'
 NAMESPACE 'dd8772e1-d792-4fa4-996b-1870577efc0d';
```

   Si vous souhaitez un contrôle plus précis de l'accès aux objets de la base de données locale, utilisez la clause WITH PERMISSIONS quand vous créez la base de données. Cela vous permet d'accorder des autorisations de niveau objet pour les objets de la base de données à l'étape 4.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF ACCOUNT
 '123456789012' NAMESPACE 'dd8772e1-d792-4fa4-996b-1870577efc0d';
```

   Vous pouvez voir les bases de données que vous avez créées à partir de l'unité de partage des données en interrogeant la vue SVV_REDSHIFT_DATABASES. Vous pouvez vous connecter directement à ces bases de données ou vous connecter à une base de données locale sur votre cluster client et effectuer une requête entre bases de données pour interroger les données des bases de données partagées. Vous ne pouvez pas créer une unité de partage des données au-dessus des objets de base de données créés à partir d'une unité de partage des données existant. Toutefois, vous pouvez copier les données dans une table distincte du cluster consommateur, effectuer le traitement nécessaire, puis partager les nouveaux objets créés.

3. (Facultatif) Créez des schémas externes pour référencer et attribuer des autorisations détaillées à des schémas spécifiques de la base de données consommateur importée sur le cluster consommateur. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL SCHEMA.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA
 'public';
```

4. Selon les besoins, accordez aux utilisateurs et rôles du cluster consommateur des autorisations sur les bases de données et références de schéma créées à partir des unités de partage des données. Pour plus d'informations, consultez GRANT ou REVOKE.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Si vous avez créé votre base de données sans la clause WITH PERMISSIONS, vous ne pouvez attribuer des autorisations sur l'ensemble de la base de données créée à partir de l'unité de partage des données qu'à vos utilisateurs ou groupes. Dans certains cas, vous avez besoin de contrôles précis sur un sous-ensemble d'objets de base de données créés à partir de l'unité de partage des données. Si tel est le cas, vous pouvez créer une référence de schéma externe tournée vers des schémas spécifiques dans l'unité de partage des données comme décrit à l'étape précédente. Vous pouvez ensuite fournir des autorisations détaillées au niveau du schéma. Vous pouvez également créer des vues à liaison tardive sur les objets partagés et les utiliser pour attribuer des autorisations détaillées. Vous pouvez également envisager que les clusters producteur créent des unités de partage des données supplémentaires pour vous avec la précision requise. Vous pouvez créer autant de références de schéma à la base de données créée à partir de l'unité de partage des données que vous le souhaitez.

Si vous avez créé votre base de données avec la clause WITH PERMISSIONS à l'étape 2, vous devez attribuer des autorisations de niveau objet dans la base de données partagée. Un utilisateur disposant uniquement de l'autorisation USAGE ne peut accéder à aucun objet d'une base de données créée avec la clause WITH PERMISSIONS tant qu'il n'a pas obtenu d'autorisations de niveau objet supplémentaires.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

5. Interroger les données dans les objets partagés dans les unités de partage des données.

   Les utilisateurs et les rôles disposant d'autorisations sur les bases de données consommateur et les schémas sur les clusters consommateur peuvent explorer et naviguer entre les métadonnées de tous les objets partagés. Ils peuvent également explorer et naviguer entre les objets locaux dans un cluster consommateur. Pour ce faire, utilisez des pilotes JDBC ou ODBC ou des vues SVV_ALL et SVV_REDSHIFT.

   Les clusters producteur peuvent avoir de nombreux schémas dans la base de données, les tables et les vues au sein de chaque schéma. Les utilisateurs du côté consommateur ne peuvent voir que le sous-ensemble d'objets mis à disposition via l'unité de partage des données. Ces utilisateurs ne peuvent pas voir l'intégralité des métadonnées du cluster producteur. Cette approche permet de fournir un contrôle détaillé de la sécurité des métadonnées avec le partage des données.

Vous continuez à vous connecter aux bases de données de cluster locales. Mais maintenant, vous pouvez également lire à partir des bases de données et des schémas qui sont créés à partir de l'unité de partage des données en utilisant la notation database.schema.table en trois parties. Vous pouvez effectuer des requêtes qui s'étendent sur toutes les bases de données qui sont visibles pour vous. Il peut s'agir de bases de données locales sur le cluster ou de bases de données créées à partir des unités de partage des données. Les clusters consommateur ne peuvent pas se connecter aux bases de données créées à partir des unités de partage des données.

Vous pouvez accéder aux données à l'aide d'une qualification complète. Pour plus d'informations, consultez Exemples de requêtes entre bases de données.

```
SELECT * FROM sales_db.public.tickit_sales_redshift;
```

Vous ne pouvez utiliser les instructions SELECT que sur les objets partagés. Toutefois, vous pouvez créer des tables dans le cluster consommateur en interrogeant les données des objets partagés dans une base de données locale différente.

En plus d'adresser des requêtes, les consommateurs peuvent créer des vues sur des objets partagés. Seules les vues à liaison tardive et les vues matérialisées sont prises en charge. Amazon Redshift ne prend pas en charge les vues standard sur les données partagées. Les vues créées par les consommateurs peuvent s'étendre sur plusieurs bases de données locales ou bases de données créées à partir d'unités de partage des données. Pour de plus amples informations, veuillez consulter CREATE VIEW.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.
CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Partage de données entre Régions AWS

Vous pouvez partager des données à des fins de lecture entre des clusters Amazon Redshift dans des Régions AWS. Grâce au partage de données entre régions, vous pouvez partager des données entre elles Régions AWS sans avoir à les copier manuellement. Vous n'avez pas besoin de décharger vos données dans Amazon S3 et de les copier dans un nouveau cluster Amazon Redshift ou d'effectuer une copie d'instantané entre régions.

Grâce au partage de données entre régions, vous pouvez partager des données entre des clusters situés dans la même région ou de manière différente Compte AWS, Comptes AWS même lorsque les clusters se trouvent dans des régions différentes. Lorsque vous partagez des données avec des clusters Amazon Redshift situés dans les mêmes clusters Compte AWS mais différents Régions AWS, suivez le même flux de travail que le partage de données au sein d'un. Compte AWS Pour de plus amples informations, veuillez consulter Partage de l'accès en lecture aux données au sein d'un Compte AWS.

Si les clusters partageant des données se situent dans Comptes AWS des entités différentes Régions AWS, vous pouvez suivre le même flux de travail que pour partager des données entre eux Comptes AWS et inclure des associations au niveau régional dans le cluster de consommateurs. Le partage de données entre régions prend en charge l'association de partage de données à l'ensemble Compte AWS, à l'intégralité ou à des espaces Région AWS de noms spécifiques au sein d'un. Région AWS Pour plus d'informations sur le partage de données entre Comptes AWS personnes, consultezPartage de données entre Comptes AWS.

Lorsqu'il consomme des données provenant d'une autre région, le consommateur paie les frais de transfert de données entre régions de la région productrice vers la région consommatrice.

Pour utiliser l'unité de partage des données, un administrateur de compte consommateur peut associer l'unité de partage des données de l'une des trois manières suivantes.

- Association avec un ensemble Compte AWS englobant tous ses Régions AWS
- Association avec un particulier Région AWS dans un Compte AWS
- Association avec des espaces de noms spécifiques au sein d'un Région AWS

Lorsque l'administrateur choisit l'intégralité Compte AWS, tous les espaces de noms existants et futurs des différents éléments Régions AWS du compte ont accès aux partages de données. Un administrateur de compte client peut également choisir des espaces de noms spécifiques Régions

AWS ou des espaces de noms au sein d'une région pour lui accorder l'accès aux partages de données.

Si vous êtes un producteur, un administrateur ou un propriétaire de base de données, créez un partage de données, ajoutez des objets de base de données et des consommateurs de données au partage de données, et accordez des autorisations aux consommateurs de données. Pour de plus amples informations, veuillez consulter actions du producteur et de l'administrateur.

Si vous êtes administrateur de compte producteur, autorisez les partages de données à l'aide de la console AWS Command Line Interface (AWS CLI) ou Amazon Redshift et choisissez les consommateurs de données.

Si vous êtes un administrateur de compte consommateur, procédez comme suit :

Pour associer un ou plusieurs partages de données partagés depuis d'autres comptes à l'ensemble Compte AWS ou à des espaces de noms spécifiques Régions AWS au sein d'un compte Région AWS, utilisez la console Amazon Redshift.

Grâce au partage de données entre régions, vous pouvez ajouter des clusters dans un environnement spécifique à Région AWS l'aide de la AWS Command Line Interface console (AWS CLI) ou Amazon Redshift.

Pour spécifier une ou plusieurs AWS régions, vous pouvez utiliser la commande `associate-data-share-consumer` CLI avec l'`consumer-region`option facultative.

Dans le cas de la CLI, l'exemple suivant `Salesshare` associe le à l'ensemble Compte AWS à l'`associate-entire-account`option. Vous ne pouvez associer qu'une seule région à la fois.

```
aws redshift associate-data-share-consumer
--region {PRODUCER_REGION}
--data-share-arn arn:aws:redshift:{PRODUCER_REGION}:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/Salesshare
--associate-entire-account
```

L'exemple suivant associe le `Salesshare` à la région USA Est (Ohio) (`us-east-2`).

```
aws redshift associate-data-share-consumer
--region {PRODUCER_REGION}
--data-share-arn arn:aws:redshift:{PRODUCER_REGION}:0123456789012:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/Salesshare
--consumer-region 'us-east-2'
```

L'exemple suivant `Salesshare` associe le à un espace de noms de consommateur spécifique Compte AWS dans un autre espace de la région Asie-Pacifique (Sydney) (`ap-southeast-2`).

```
aws redshift associate-data-share-consumer
--data-share-arn arn:aws:redshift:{PRODUCER_REGION}:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/Salesshare
--consumer-arn 'arn:aws:redshift:ap-southeast-2:{CONSUMER_ACCOUNT}:namespace:
{ConsumerImmutableClusterId}'
```

Vous pouvez utiliser la console Amazon Redshift pour associer des partages de données à vos espaces de noms complets Compte AWS Régions AWS ou spécifiques au sein d'un. Région AWS Pour ce faire, connectez-vous au https://console.aws.amazon.com/redshiftv2/. Associez ensuite un ou plusieurs partages de données partagés depuis d'autres comptes à l'intégralité de votre espace de noms Compte AWS, à l'intégralité Région AWS ou à un espace de noms spécifique au sein d'un. Région AWS Pour de plus amples informations, veuillez consulter Associer un partage de données à un autre Compte AWS dans Amazon Redshift.

Une fois que les espaces de noms Compte AWS ou des espaces de noms spécifiques sont associés, les partages de données peuvent être utilisés. Vous pouvez également modifier l'association d'unité de partage des données à tout moment. Lorsque vous remplacez l'association d'espaces de noms individuels par un Compte AWS, Amazon Redshift remplace les espaces de noms par les informations. Compte AWS Lorsque l'association passe d'un espace de noms Compte AWS à un espace de noms spécifique, Amazon Redshift remplace les informations par Compte AWS les informations de l'espace de noms. Lorsque vous passez d'une association complète Compte AWS à des AWS régions et espaces de noms spécifiques, Amazon Redshift remplace les informations par les informations relatives à Compte AWS la région et à l'espace de noms spécifiques.

Si vous êtes un administrateur client, vous pouvez créer des bases de données locales qui font référence aux partages de données et accorder des autorisations sur les bases de données créées à partir des partages de données aux utilisateurs ou aux rôles du cluster de consommateurs selon les besoins. Vous pouvez aussi créer des vues sur des objets partagés et créer des schémas externes pour référencer et attribuer des autorisations détaillées à des schémas spécifiques de la base de données consommateur importée sur le cluster consommateur. Pour plus d'informations, consultez actions de l'administrateur du consommateur.

Gestion du contrôle des coûts pour le partage de données entre régions

Avec Amazon Redshift, vous pouvez gérer le contrôle des coûts liés au partage de données entre régions en configurant le partage de données afin de limiter la quantité de données transférées

entre les régions. AWS La gestion du contrôle des coûts pour le partage de données entre régions vous permet de définir des limites de transfert de données, de surveiller l'utilisation des transferts de données et de recevoir des notifications lorsque vous approchez ou dépassez ces limites.

Lors de la consommation de données provenant d'une autre région, le consommateur paie les frais de transfert de données entre régions de la région productrice vers la région consommatrice. Le prix du transfert de données est différent selon les régions. Les frais sont basés sur les octets de données analysés pour chaque exécution de requête réussie. Pour plus d'informations sur la tarification Amazon Redshift, consultez Tarification Amazon Redshift.

Vous êtes facturé pour le nombre d'octets, arrondi au mégaoctet suivant, avec un minimum de 10 Mo par requête. Vous pouvez définir des contrôles de coûts sur l'utilisation de vos requêtes et afficher la quantité de données transférées par requête sur votre cluster.

Pour surveiller et contrôler votre utilisation et les coûts d'utilisation associés du partage de données entre régions, vous pouvez créer des limites d'utilisation quotidiennes, hebdomadaires et mensuelles, ainsi que définir des actions exécutées automatiquement par Amazon Redshift si ces limites sont atteintes afin de maintenir votre budget avec prévisibilité.

Selon les limites d'utilisation que vous définissez, les actions entreprises par Amazon Redshift peuvent consister à enregistrer un événement dans une table système, à envoyer une CloudWatch alarme et à avertir un administrateur via un Amazon SNS, ou à désactiver le partage de données entre régions pour une utilisation ultérieure.

Pour créer des limites d'utilisation dans la console Amazon Redshift, sélectionnez Configure usage limit (Configurer la limite d'utilisation) sous Actions pour votre cluster. Vous pouvez surveiller vos tendances d'utilisation et recevoir des alertes en cas d'utilisation dépassant les limites définies grâce à des CloudWatch mesures générées automatiquement à partir des onglets Performances du cluster ou Surveillance. Vous pouvez également créer, modifier et supprimer des limites d'utilisation par programmation à l'aide des opérations de l'API Amazon Redshift AWS CLI ou de l'API Amazon Redshift.

Partage de données Amazon Redshift sous licence sur AWS Data Exchange

Lors de la création de AWS Data Exchange partages de données et de leur ajout à un AWS Data Exchange produit, les fournisseurs peuvent octroyer des licences sur des données dans Amazon Redshift que les consommateurs peuvent découvrir, consulter et consulter sur Amazon up-to-date Redshift lorsqu'ils ont des abonnements actifs. AWS Data Exchange

Lorsque des AWS Data Exchange partages de données sont ajoutés à un AWS Data Exchange produit, les consommateurs ont automatiquement accès aux partages de données d'un produit lorsque leur abonnement commence et conservent leur accès tant que leur abonnement est actif.

Rubriques

- Utilisation des partages AWS Data Exchange de données en tant que producteur
- Travailler avec des AWS Data Exchange partages de données en tant que consommateur

Utilisation des partages AWS Data Exchange de données en tant que producteur

Avec Amazon Redshift, vous pouvez partager des produits de données en temps réel en AWS Data Exchange tant que producteur en créant et en gérant des partages de données.

Si vous êtes un administrateur producteur, suivez ces étapes pour gérer les partages AWS Data Exchange de données sur la console Amazon Redshift :

1. Créez des partages de données dans votre cluster pour partager des données AWS Data Exchange et autoriser l'accès AWS Data Exchange à ces partages de données.

   Les super-utilisateurs de cluster et les propriétaires de base de données peuvent créer des datashares. Chaque datashare est associé à une base de données lors de sa création. Seuls les objets de cette base de données peuvent être partagés dans ce datashare. Plusieurs datashares peuvent être créés sur la même base de données avec la même précision d'objets ou une précision différente. Il n’y a pas de limite au nombre d’unités de partage des données que vous pouvez créer sur un cluster.

   Vous pouvez également créer des datashares via la console Amazon Redshift. Pour de plus amples informations, veuillez consulter Création d'un partage de données.

   Utilisez l'option MANAGEDBY ADX pour accorder implicitement l'accès au partage de données AWS Data Exchange lors de l'exécution de l'instruction CREATE DATASHARE. Cela indique que ce partage de données est AWS Data Exchange géré. Vous ne pouvez utiliser l’option MANAGEDBY ADX que lorsque vous créez une unité de partage des données. Vous ne pouvez pas utiliser l’instruction ALTER DATASHARE pour modifier une unité de partage des données existante afin d’ajouter l’option MANAGEDBY ADX. Une fois qu’un partage de données est créé avec l’option MANAGEDBY ADX, seul AWS Data Exchange peut accéder à l’unité de partage des données et la gérer.

```
CREATE DATASHARE salesshare
[[SET] MANAGEDBY [=] {ADX} ];
```

2. Ajouter des objets aux unités de partage des données. L'administrateur du producteur continue de gérer les objets de partage de données disponibles dans un AWS Data Exchange partage de données.

   Pour ajouter des objets à une unité de partage des données, ajoutez le schéma avant d'ajouter des objets. Lorsque vous ajoutez un schéma, Amazon Redshift n'ajoute pas tous les objets qu'il contient. Vous devez les ajouter explicitement. Pour plus d'informations, consultez ALTER DATASHARE.

   ```
   ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
   ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
   ```

   Vous pouvez également ajouter des vues à une unité de partage des données.

   ```
   CREATE VIEW public.sales_data_summary_view AS SELECT * FROM
    public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD TABLE public.sales_data_summary_view;
   ```

   Utilisez ALTER DATASHARE pour partager des schémas, des tables, des vues et des fonctions dans un schéma donné. Les super-utilisateurs, les propriétaires d'unité de partage des données ou les utilisateurs disposant des autorisations ALTER ou ALL sur l'unité de partage des données peuvent modifier l'unité de partage des données pour y ajouter des objets ou en supprimer. Les utilisateurs doivent avoir les autorisations nécessaires pour ajouter ou supprimer des objets de l'unité de partage des données. Les utilisateurs doivent également être les propriétaires des objets ou disposer des autorisations SELECT, USAGE ou ALL sur les objets.

   Utilisez la clause INCLUDENEW pour ajouter au partage de données les nouvelles tables, vues ou fonctions SQL définies par l'utilisateur (UDFs) créées dans un schéma spécifié. Seuls les super-utilisateurs peuvent modifier cette propriété pour chaque paire datashare-schéma.

   ```
   ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
   ALTER DATASHARE salesshare SET INCLUDENEW = TRUE FOR SCHEMA PUBLIC;
   ```

Vous pouvez également utiliser la console Amazon Redshift pour ajouter ou supprimer des objets des unités de partage des données. Pour plus d'informations, consultez Ajouter des objets de partage de données aux partages de données, Supprimer des objets d'unité de partage des données des unités de partage des données et Modification de partages AWS Data Exchange de données.

3. Pour autoriser l'accès aux partages de données pour AWS Data Exchange, effectuez l'une des opérations suivantes :

   - Autorisez explicitement l'accès au partage de données AWS Data Exchange en utilisant le ADX mot clé dans l'`aws redshift authorize-data-share`API. Cela permet AWS Data Exchange de reconnaître le partage de données dans le compte de service et de gérer l'association des consommateurs au partage de données.

     ```
     aws redshift authorize-data-share
     --data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
     {PRODUCER_CLUSTER_NAMESPACE}/salesshare
     --consumer-identifier ADX
     ```

     Vous pouvez utiliser une clé conditionnelle `ConsumerIdentifier` pour autoriser ou `DeauthorizeDataShare` APIs refuser explicitement d'appeler AWS Data Exchange les deux APIs dans le cadre de la politique IAM. `AuthorizeDataShare`

     ```
     {
         "Version": "2012-10-17",
         "Statement": [
             {
                 "Sid": "VisualEditor0",
                 "Effect": "Deny",
                 "Action": [
                     "redshift:AuthorizeDataShare",
                     "redshift:DeauthorizeDataShare"
                 ],
                 "Resource": "*",
                 "Condition": {
                     "StringEqualsIgnoreCase": {
                         "redshift:ConsumerIdentifier": "ADX"
                     }
                 }
             }
     ```

```
    ]
}
```

- Utilisez la console Amazon Redshift pour autoriser ou supprimer l'autorisation d'unités de partage des données AWS Data Exchange . Pour de plus amples informations, veuillez consulter Autoriser ou supprimer l'autorisation des unités de partage des données.
- Vous pouvez éventuellement autoriser implicitement l'accès au partage de AWS Data Exchange données lors de l'importation du partage de données dans un ensemble de données. AWS Data Exchange

Pour supprimer l'autorisation d'accès aux partages de AWS Data Exchange données, utilisez le ADX mot clé dans l'opération d'`aws redshift deauthorize-data-share`API. Ainsi, vous autorisez AWS Data Exchange à reconnaître l'unité de partage des données dans le compte de service et de gérer l'association de suppression depuis l'unité de partage des données.

```
aws redshift deauthorize-data-share
--data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/salesshare
--consumer-identifier ADX
```

4. Répertorie les unités de partage des données créées dans le cluster et examine le contenu de l'unité de partage des données.

   L'exemple suivant montre comment afficher les informations d'une unité de partage des données nommée salesshare. Pour plus d'informations, consultez DESC DATASHARE et SHOW DATASHARES.

```
DESC DATASHARE salesshare;

 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name          |   include_new
------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+-------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_users_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_venue_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_category_redshift|
```

```
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_date_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_event_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_listing_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_sales_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | schema      | public                         |  t
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | view        | public.sales_data_summary_view |
```

L'exemple suivant montre comment afficher les datashares sortants dans un cluster producteur.

```
SHOW DATASHARES LIKE 'sales%';
```

La sortie ressemble à ce qui suit.

```
share_name | share_owner  | source_database | consumer_database | share_type |
 createdate      | is_publicaccessible  | share_acl | producer_account |
 producer_namespace
-----------+--------------+-----------------+-------------------+------------
+---------------------+----------------------+-----------+-----------------
+--------------------------------------
salesshare |    100       | dev             |                   |  OUTBOUND
 | 2020-12-09 02:27:08 |          True        |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

Pour plus d'informations, consultez DESC DATASHARE et SHOW DATASHARES.

Vous pouvez également utiliser SVV_DATASHARES, SVV_DATASHARE_CONSUMERS, et SVV_DATASHARE_OBJECTS pour afficher les datashares, les objets du datashare et les consommateurs de datashare.

5. Supprimer des unités de partage des données. Nous vous recommandons de ne pas supprimer un partage de AWS Data Exchange données partagé avec d'autres personnes à Comptes AWS l'aide de l'instruction DROP DATASHARE. Ces comptes perdront l'accès à l'unité de partage des données. Cette action est irréversible. Cela pourrait enfreindre les conditions de l'offre de produits de données AWS Data Exchange. Si vous souhaitez supprimer un partage de AWS Data Exchange données, consultez. Note d'utilisation de DROP DATASHARE

L’exemple suivant montre la suppression d’une unité de partage des données appelé salesshare.

```
DROP DATASHARE salesshare;
ERROR:  Drop of ADX-managed datashare salesshare requires session variable
 datashare_break_glass_session_var to be set to value '620c871f890c49'
```

Pour autoriser la suppression d'un partage de données, définissez la AWS Data Exchange variable datashare_break_glass_session_var et réexécutez l'instruction DROP DATASHARE. Si vous souhaitez supprimer un partage de AWS Data Exchange données, consultez. Note d’utilisation de DROP DATASHARE

Vous pouvez également utiliser la console Amazon Redshift pour supprimer des unités de partage des données. Pour plus d'informations, consultez Supprimer les AWS Data Exchange partages de données créés dans votre compte.

6. Utilisez ALTER DATASHARE pour supprimer des objets des datashares à n'importe quel point du datashare. Utilisez REVOKE USAGE ON pour révoquer les autorisations de certains consommateurs sur l’unité de partage des données. Il révoque les autorisations USAGE sur les objets dans une unité de partage des données et arrête instantanément l’accès à tous les clusters consommateur. Répertorier des unités de partage des données et des requêtes de métadonnées, telles que la liste des bases de données et des tables, ne renvoie pas les objets partagés après la révocation de l’accès.

   ```
   ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
   ```

   Vous pouvez également utiliser la console Amazon Redshift pour modifier les datashares. Pour de plus amples informations, veuillez consulter Modification de partages AWS Data Exchange de données.

7. Accordez ou révoquez GRANT USAGE pour les partages de AWS Data Exchange données. Vous ne pouvez pas accorder ou révoquer GRANT USAGE pour le partage de AWS Data Exchange données. L'exemple suivant montre une erreur lorsque l'autorisation GRANT USAGE est accordée à un Compte AWS datashare qui AWS Data Exchange gère.

   ```
   CREATE DATASHARE salesshare MANAGEDBY ADX;
   ```

   ```
   GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '012345678910';
   ```

```
ERROR:  Permission denied to add/remove consumer to/from datashare salesshare.
 Datashare consumers are managed by ADX.
```

Pour plus d'informations, consultez GRANT ou REVOKE.

Si vous êtes administrateur producteur, procédez comme suit pour créer et publier un produit de partage de données sur la AWS Data Exchange console :

- Lorsque le AWS Data Exchange partage de données a été créé, le producteur crée un nouvel ensemble de données, importe des actifs, crée une révision, puis crée et publie un nouveau produit.

  Utilisez la console Amazon Redshift pour créer des jeux de données. Pour de plus amples informations, veuillez consulter Création d'ensembles de données sur AWS Data Exchange.

  Pour plus d'informations, voir Fournir des produits de données sur AWS Data Exchange.

Travailler avec des AWS Data Exchange partages de données en tant que consommateur

Avec Amazon Redshift, vous pouvez accéder à des ensembles de données et les analyser AWS Data Exchange sans avoir à stocker ou à gérer des copies des données.

Si vous êtes un consommateur, suivez ces étapes pour découvrir les produits de données contenant des partages de AWS Data Exchange données et interroger les données Amazon Redshift :

1. Sur la AWS Data Exchange console, découvrez les produits de données contenant des partages de données et abonnez-vous à ces produits AWS Data Exchange .

   Une fois votre abonnement lancé, vous pouvez accéder aux données Amazon Redshift sous licence qui sont importées sous forme d'actifs dans des ensembles de données contenant des partages de données. AWS Data Exchange

   Pour plus d'informations sur la façon de commencer à utiliser des produits de données contenant des partages de AWS Data Exchange données, voir Abonnement à des produits de données sur. AWS Data Exchange
2. Sur la console Amazon Redshift, créez un cluster Amazon Redshift, si nécessaire.

   Pour plus d'informations sur la création d'un cluster, consultez Création d'un cluster.

3. Répertorier les unités de partage des données mises à votre disposition et afficher leur contenu. Pour plus d'informations, consultez DESC DATASHARE et SHOW DATASHARES.

   L'exemple suivant montre comment afficher les informations des unités de partage des données entrants d'un espace de noms producteur spécifié. Lorsque vous exécutez DESC DATASHARE en tant qu'administrateur client, vous devez spécifier les options ACCOUNT et NAMESPACE pour afficher les partages de données entrants.

```
DESC DATASHARE salesshare of ACCOUNT '123456789012' NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';

 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name           |   include_new
------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_users_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_venue_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_category_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_date_redshift     |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_event_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_listing_redshift  |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | table       | public.tickit_sales_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | schema      | public                          |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare
 | view        | public.sales_data_summary_view  |
```

   Seuls les super-utilisateurs de cluster peuvent le faire. Vous pouvez également utiliser SVV_DATASHARES pour afficher les unités de partage des données et SVV_DATASHARE_OBJECTS pour afficher les objets dans l'unité de partage des données.

   L'exemple suivant affiche les datashares entrants dans un cluster consommateur.

```
SHOW DATASHARES LIKE 'sales%';
```

```
 share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
------------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+--------------------------------------
 salesshare |             |                 |                   | INBOUND
    |            |         t           |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

4. Créez des bases de données locales qui font référence aux unités de partage des données. Vous devez spécifier les options ACCOUNT et NAMESPACE pour créer des bases de données locales pour des unités de partage des données AWS Data Exchange . Pour de plus amples informations, veuillez consulter CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF ACCOUNT '123456789012'
 NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Si vous souhaitez un contrôle plus précis de l'accès aux objets de la base de données locale, utilisez la clause WITH PERMISSIONS quand vous créez la base de données. Cela vous permet d'accorder des autorisations de niveau objet pour les objets de la base de données à l'étape 6.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF ACCOUNT
 '123456789012' NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Vous pouvez voir les bases de données que vous avez créées à partir de l'unité de partage des données en interrogeant la vue SVV_REDSHIFT_DATABASES. Vous pouvez vous connecter directement à ces bases de données ou vous connecter à une base de données locale sur votre cluster client et effectuer une requête entre bases de données pour interroger les données des bases de données partagées. Vous ne pouvez pas créer une unité de partage des données au-dessus des objets de base de données créés à partir d'une unité de partage des données existant. Toutefois, vous pouvez copier les données dans une table distincte du cluster consommateur, effectuer le traitement nécessaire, puis partager les nouveaux objets créés.

Vous pouvez également utiliser la console Amazon Redshift pour créer des bases de données à partir des unités de partage des données. Pour plus d'informations, consultez Créer des bases de données à partir d'unités de partage des données.

5. (Facultatif) Créez des schémas externes pour faire référence et attribuer des autorisations détaillées à des schémas spécifiques dans la base de données consommateur importée sur le cluster consommateur. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL SCHEMA.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA
 'public';
```

6. Selon les besoins, accordez aux utilisateurs et rôles du cluster consommateur des autorisations sur les bases de données et références de schéma créées à partir des unités de partage des données. Pour plus d'informations, consultez GRANT ou REVOKE.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Si vous avez créé votre base de données sans la clause WITH PERMISSIONS, vous ne pouvez attribuer des autorisations sur l'ensemble de la base de données créée à partir de l'unité de partage des données qu'à vos utilisateurs et groupes. Dans certains cas, vous avez besoin de contrôles précis sur un sous-ensemble d'objets de base de données créés à partir de l'unité de partage des données. Si tel est le cas, vous pouvez créer une référence de schéma externe tournée vers des schémas spécifiques dans l'unité de partage des données (comme décrit à l'étape précédente) et fournir des autorisations détaillées au niveau du schéma.

Vous pouvez également créer des vues à liaison tardive sur les objets partagés et les utiliser pour attribuer des autorisations détaillées. Vous pouvez également envisager que les clusters producteur créent des unités de partage des données supplémentaires pour vous avec la précision requise. Vous pouvez créer autant de références de schéma à la base de données créée à partir de l'unité de partage des données que vous le souhaitez.

Si vous avez créé votre base de données avec la clause WITH PERMISSIONS à l'étape 4, vous devez attribuer des autorisations de niveau objet dans la base de données partagée. Un utilisateur disposant uniquement de l'autorisation USAGE ne peut accéder à aucun objet d'une base de données créée avec la clause WITH PERMISSIONS tant qu'il n'a pas obtenu d'autorisations de niveau objet supplémentaires.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

7. Interroger les données dans les objets partagés dans les unités de partage des données.

   Les utilisateurs et les rôles disposant d'autorisations sur les bases de données consommateur et les schémas sur les clusters consommateur peuvent explorer et naviguer entre les métadonnées de tous les objets partagés. Ils peuvent également explorer et naviguer entre les objets locaux dans un cluster consommateur. Pour ce faire, ils utilisent des pilotes JDBC ou ODBC, les commandes SHOW ou les vues SVV_ALL et SVV_REDSHIFT.

   Les clusters producteur peuvent avoir de nombreux schémas dans la base de données, les tables et les vues au sein de chaque schéma. Les utilisateurs du côté consommateur ne peuvent voir que le sous-ensemble d'objets mis à disposition via l'unité de partage des données. Ces utilisateurs ne peuvent pas voir l'intégralité des métadonnées du cluster producteur. Cette approche permet de fournir un contrôle détaillé de la sécurité des métadonnées avec le partage des données.

   Vous continuez à vous connecter aux bases de données de cluster locales. Mais maintenant, vous pouvez également lire à partir des bases de données et des schémas qui sont créés à partir de l'unité de partage des données en utilisant la notation database.schema.table en trois parties. Vous pouvez effectuer des requêtes qui s'étendent sur toutes les bases de données qui sont visibles pour vous. Il peut s'agir de bases de données locales sur le cluster ou de bases de données créées à partir des unités de partage des données. Vous pouvez également vous connecter directement à ces bases de données de consommateurs et exécuter des requêtes sur les objets partagés avec une notation partielle.

   Vous pouvez accéder aux données à l'aide d'une qualification complète. Pour plus d'informations, consultez Exemples de requêtes entre bases de données.

```
SELECT * FROM sales_db.public.tickit_sales_redshift ORDER BY 1,2 LIMIT 5;

 salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime
---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------
       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |    728.00 |
   109.20 | 2008-02-18 02:36:48
       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |     76.00 |
    11.40 | 2008-06-06 05:00:16
       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |    350.00 |
    52.50 | 2008-06-06 08:26:17
```

```
   4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |    175.00 |
26.25 | 2008-06-09 08:38:52
   5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |    154.00 |
23.10 | 2008-08-31 09:17:02
```

Vous ne pouvez utiliser les instructions SELECT que sur les objets partagés. Toutefois, vous pouvez créer des tables dans le cluster consommateur en interrogeant les données des objets partagés dans une base de données locale différente.

En plus des requêtes, les consommateurs peuvent créer des vues sur des objets partagés. Seules les vues à liaison tardive ou les vues matérialisées sont prises en charge. Amazon Redshift ne prend pas en charge les vues standard sur les données partagées. Les vues créées par les consommateurs peuvent s'étendre sur plusieurs bases de données locales ou bases de données créées à partir d'unités de partage des données. Pour de plus amples informations, veuillez consulter CREATE VIEW.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.
CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Commencer à utiliser les AWS Lake Formation partages de données gérés

Avec Amazon Redshift, vous pouvez accéder aux données en temps réel et les partager entre différents AWS comptes et clusters AWS Lake Formation Amazon Redshift via des partages de données gérés. AWS Lake Formation les partages de données permettent aux fournisseurs de données de partager en toute sécurité les données en temps réel de leur lac de données Amazon S3 avec n'importe quel consommateur, y compris les autres AWS comptes et les clusters Amazon Redshift.

Utilisation d'unités de partage des données gérées par Lake Formation en tant que producteur

Avec Amazon Redshift, vous pouvez accéder aux données partagées par le biais AWS Lake Formation de partages de données et les analyser. AWS Lake Formation les partages de données

permettent un partage sécurisé des données entre les AWS comptes et les clusters Amazon Redshift sans avoir à copier ou à déplacer les données sous-jacentes.

Le partage de données vous AWS Lake Formation permet de définir de manière centralisée AWS Lake Formation les autorisations des partages de données Amazon Redshift et de restreindre l'accès des utilisateurs aux objets d'un partage de données.

Avec Amazon Redshift, vous pouvez partager en toute sécurité des données en temps réel entre des AWS comptes et des clusters Amazon Redshift en AWS Lake Formation utilisant des partages de données gérés en tant que producteur. Un partage de données géré par Lake Formation est un objet qui vous permet de partager des données en direct depuis votre cluster Amazon Redshift avec d'autres comptes et services. AWS

En tant qu'administrateur de cluster producteur ou de groupe de travail, procédez comme suit pour partager des unités de partage des données avec Lake Formation :

1. Créez des partages de données dans votre cluster et autorisez l'accès AWS Lake Formation aux partages de données.

   Seuls les super-utilisateurs de cluster et les propriétaires de base de données peuvent créer des unités de partage des données. Chaque datashare est associé à une base de données lors de sa création. Seuls les objets de cette base de données peuvent être partagés dans ce datashare. Plusieurs datashares peuvent être créés sur la même base de données avec la même précision d'objets ou une précision différente. Il n'y a pas de limite au nombre d'unités de partage des données que vous pouvez créer sur un cluster.

   ```
   CREATE DATASHARE salesshare;
   ```

2. Ajoutez des objets à l'unité de partage des données. L'administrateur du cluster producteur ou du groupe de travail continue de gérer les objets de l'unité de partage des données disponibles. Pour ajouter des objets à une unité de partage des données, ajoutez le schéma avant d'ajouter des objets. Lorsque vous ajoutez un schéma, Amazon Redshift n'ajoute pas tous les objets qu'il contient. Vous devez les ajouter explicitement. Pour plus d'informations, consultez ALTER DATASHARE.

   ```
   ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
   ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
   ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
   ```

Vous pouvez également ajouter des vues à une unité de partage des données. Seules les vues standard, à liaison tardive et matérialisées sont prises en charge.

```
CREATE VIEW public.sales_data_summary_view AS SELECT * FROM
 public.tickit_sales_redshift;
ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
```

Utilisez ALTER DATASHARE pour partager des schémas, des tables et des vues dans un schéma donné. Les super-utilisateurs, les propriétaires d'unité de partage des données ou les utilisateurs disposant des autorisations ALTER ou ALL sur l'unité de partage des données peuvent modifier l'unité de partage des données pour y ajouter des objets ou en supprimer. Les utilisateurs de la base de données doivent également être les propriétaires des objets ou disposer des autorisations SELECT, USAGE ou ALL sur les objets.

Utilisez la clause INCLUDENEW pour ajouter de nouvelles tables et des vues créées dans un schéma spécifié à l'unité de partage des données. Seuls les super-utilisateurs peuvent modifier cette propriété pour chaque paire datashare-schéma.

```
ALTER DATASHARE salesshare ADD SCHEMA PUBLIC;
ALTER DATASHARE salesshare SET INCLUDENEW = TRUE FOR SCHEMA PUBLIC;
```

3. Autorisez l'unité de partage des données à accéder à un compte administrateur Lake Formation.

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '012345678910' VIA DATA CATALOG;
```

Pour annuler l'utilisation, utilisez la commande suivante.

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '012345678910' VIA DATA CATALOG;
```

4. Autorisez l'unité de partage des données à accéder à Lake Formation à l'aide de l'opération d'API `aws redshift authorize-data-share`. Cela permet à Lake Formation de reconnaître l'unité de partage des données dans le compte de service et de gérer l'association des consommateurs à l'unité de partage des données.

```
aws redshift authorize-data-share
--data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/salesshare
--consumer-identifier {"DataCatalog/<consumer-account-id>"}
```

Pour supprimer l'autorisation des unités de partage des données gérés par Lake Formation, utilisez l'opération d'API `aws redshift deauthorize-data-share`. Ce faisant, vous autorisez AWS Lake Formation à reconnaître le partage de données dans le compte de service et à supprimer l'autorisation.

```
aws redshift deauthorize-data-share
--data-share-arn arn:aws:redshift:us-east-1:{PRODUCER_ACCOUNT}:datashare:
{PRODUCER_CLUSTER_NAMESPACE}/salesshare
--consumer-identifier {"DataCatalog/<consumer-account-id>"}
```

À tout moment, si l'administrateur du cluster producteur ou du groupe de travail décide qu'il n'est plus nécessaire de partager des données avec le cluster consommateur ou le groupe de travail, il peut utiliser DROP DATASHARE pour supprimer l'unité de partage des données, annuler les autorisations de l'unité de partage des données ou révoquer les autorisations de l'unité de partage des données. Les autorisations et les objets associés dans Lake Formation ne sont pas automatiquement supprimés.

```
DROP DATASHARE salesshare;
```

Après avoir autorisé le compte Lake Formation à gérer le partage de données, l'administrateur de Lake Formation peut découvrir le partage de données partagé, associer le partage de données à un ARN du catalogue de données et créer une base de données dans le lien vers le partage de données. AWS Glue Data Catalog Pour associer des partages de données à l'aide de AWS CLI, utilisez la commande. associate-data-share-consumer Pour partager une unité de partage des données entre des Régions AWS, spécifiez le paramètre `--region` dans la commande `associate-data-share-consumer` ou utilisez la console AWS pour choisir vos consommateurs de données. L'exemple suivant montre comment partager un partage de données géré par Lake Formation entre plusieurs régions.

```
aws redshift associate-data-share-consumer --region <region-1>
--data-share-arn 'arn:aws:redshift:us-
east-1:12345678912:datashare:035c45ea-61ce-86f0-8b75-19ac6102c3b7/sample_share'
--consumer-arn 'arn:aws:glue:<region-1>:111912345678:catalog'
```

L'administrateur Lake Formation doit également créer des ressources locales qui définissent la manière dont les objets de l'unité de partage des données doivent être mappés aux objets Lake Formation. Pour plus d'informations sur la découverte des unités de partage des données et la

création de ressources locales, consultez Gestion des autorisations pour les données dans une unité de partage des données Amazon Redshift.

Utilisation d'unités de partage des données gérées par Lake Formation en tant que consommateur

Avec Amazon Redshift, vous pouvez accéder aux données partagées avec vous par le biais AWS Lake Formation de partages de données et les analyser. Un partage de données est un produit de données qui contient un ensemble d'objets de données, tels que des tables ou des bases de données, provenant de différentes sources de données.

Une fois AWS Glue Data Catalog que l' AWS Lake Formation administrateur a découvert l'invitation de partage de données et créé une base de données liée au partage de données, l'administrateur du cluster de consommateurs ou du groupe de travail peut associer le cluster au partage de données et à la base de données du AWS Glue Data Catalog, créer une base de données locale pour le cluster de consommateurs ou le groupe de travail, et accorder l'accès aux utilisateurs et aux rôles du cluster de consommateurs ou du groupe de travail Amazon Redshift pour commencer à interroger. Suivez ces étapes pour configurer les autorisations d'interrogation.

1. Dans la console Amazon Redshift, créez un cluster Amazon Redshift qui fera office de cluster consommateur de groupe de travail, le cas échéant. Pour plus d'informations sur la création d'un cluster, consultez Création d'un cluster.
2. Pour répertorier les bases de données du cluster AWS Glue Data Catalog consommateur ou du groupe de travail auxquelles les utilisateurs ont accès, exécutez la commande SHOW DATABASES.

   ```
   SHOW DATABASES FROM DATA CATALOG [ACCOUNT <account-id>,<account-id2>] [LIKE
    <expression>]
   ```

   Cela permet de répertorier les ressources disponibles dans le catalogue de données, telles que l'ARN de la AWS Glue base de données, le nom de la base de données et les informations sur le partage de données.
3. À l'aide de l'ARN de la AWS Glue base de données de SHOW DATABASES, créez une base de données locale dans le cluster de consommateurs ou le groupe de travail. Pour plus d'informations, consultez CREATE DATABASE.

   ```
   CREATE DATABASE lf_db FROM ARN <lake-formation-database-ARN> WITH [NO] DATA CATALOG
    SCHEMA [<schema>];
   ```

4. Accordez l'accès aux bases de données et aux références de schéma créées à partir des unités de partage des données aux utilisateurs et rôles du cluster consommateur ou du groupe de travail selon les besoins. Pour plus d'informations, consultez GRANT ou REVOKE. Notez que les utilisateurs créés à l'aide de la commande CREATE USER ne peuvent pas accéder aux objets de l'unité de partage des données qui ont été partagés avec Lake Formation. Seuls les utilisateurs ayant accès à la fois à Redshift et à Lake Formation peuvent accéder aux unités de partage des données qui ont été partagées avec Lake Formation.

   ```
   GRANT USAGE ON DATABASE sales_db TO IAM:Bob;
   ```

   En tant qu'administrateur de cluster consommateur ou de groupe de travail, vous ne pouvez attribuer des autorisations sur l'ensemble de la base de données créée à partir de l'unité de partage des données qu'à vos utilisateurs et groupes. Dans certains cas, vous avez besoin de contrôles précis sur un sous-ensemble d'objets de base de données créés à partir de l'unité de partage des données.

   Vous pouvez également créer des vues à liaison tardive sur les objets partagés et les utiliser pour attribuer des autorisations détaillées. Vous pouvez également envisager que les clusters producteur ou les groupes de travails créent des unité de partage des données supplémentaires pour vous avec la précision requise. Vous pouvez créer autant de références de schéma à la base de données créée à partir de l'unité de partage des données.

5. Les utilisateurs de base de données peuvent utiliser les vues SVV_EXTERNAL_TABLES et SVV_EXTERNAL_COLUMNS pour rechercher toutes les tables ou colonnes partagées dans la base de données AWS Glue .

   ```
   SELECT * from svv_external_tables WHERE redshift_database_name = 'lf_db';

   SELECT * from svv_external_columns WHERE redshift_database_name = 'lf_db';
   ```

6. Interroger les données dans les objets partagés dans les unités de partage des données.

   Les utilisateurs et les rôles disposant d'autorisations sur les bases de données consommateur et les schémas sur les clusters consommateur peuvent explorer et naviguer entre les métadonnées de tous les objets partagés. Ils peuvent également explorer les objets locaux dans un cluster consommateur ou un groupe de travail. Pour ce faire, ils peuvent utiliser les pilotes JDBC ou ODBC ou les vues SVV_ALL et SVV_EXTERNAL.

```
SELECT * FROM lf_db.schema.table;
```

Vous ne pouvez utiliser les instructions SELECT que sur les objets partagés. Toutefois, vous pouvez créer des tables dans le cluster consommateur en interrogeant les données des objets partagés dans une base de données locale différente.

```
// Connect to a local cluster database

// Create a view on shared objects and access it.

CREATE VIEW sales_data
AS SELECT *
FROM sales_db.public.tickit_sales_redshift
WITH NO SCHEMA BINDING;

SELECT * FROM sales_data;
```

Commencer à utiliser les écritures dans plusieurs entrepôts à l'aide du partage de données dans Amazon Redshift

Vous pouvez partager des objets de base de données en lecture et en écriture entre différents clusters Amazon Redshift ou groupes de travail Amazon Redshift Serverless au sein d'un Compte AWS même cluster, d'un compte à l'autre et d'une région à l'autre. Les procédures décrites dans cette rubrique montrent comment configurer le partage de données incluant les autorisations d'écriture. Vous pouvez accorder des autorisations telles que SELECT, INSERT et UPDATE pour différentes tables et USAGE et CREATE pour les schémas.

Les données sont actives et accessibles à tous les entrepôts dès que vous validez une transaction d'écriture. Les administrateurs du compte du producteur peuvent déterminer si des espaces de noms ou des régions spécifiques sont accessibles en lecture seule ou s'ils ont accès aux données. read-and-write Les procédures supposent que vous travaillez sur une base de données dans un cluster provisionné ou un groupe de travail Amazon Redshift sans serveur.

Avec Amazon Redshift, vous pouvez gérer le partage de données par écriture à l'aide de la console ou de l'interface SQL pour contrôler l'accès et gouverner les données sur les clusters et les comptes

Amazon Redshift. AWS Les sections suivantes fournissent des step-by-step instructions sur la configuration et la gestion du partage de données avec des écritures à l'aide d'Amazon Redshift.

Pour obtenir la liste des régions où le partage de données est disponible, voirRégions AWS où le partage de données est disponible. Pour les considérations et les limites relatives aux écritures, voirConsidérations relatives au partage de données dans Amazon Redshift.

> **Note**
>
> Les écritures multi-entrepôts Amazon Redshift utilisant le partage de données ne sont prises en charge que sur le correctif 186 d'Amazon Redshift pour les clusters provisionnés sur la version 1.0.78881 ou supérieure, et pour les groupes de travail Amazon Redshift Serverless sur la version 1.0.78890 ou supérieure.

Rubriques

- Connexion à une base de données dans Amazon Redshift
- Actions du producteur pour les nouveaux partages de données dans Amazon Redshift
- Actions des consommateurs en faveur des nouveaux partages de données dans Amazon Redshift
- Actions du producteur pour les partages de données existants dans Amazon Redshift
- Actions des consommateurs concernant les partages de données existants dans Amazon Redshift

Connexion à une base de données dans Amazon Redshift

Avec Amazon Redshift, vous pouvez établir une connexion à votre cluster d'entrepôts de données et exécuter des requêtes SQL, charger des données ou effectuer des tâches administratives. La connexion à une base de données fait référence au processus de création d'un canal sécurisé entre une application ou un outil client et le cluster Amazon Redshift. Les sections suivantes fournissent des step-by-step instructions sur la manière de se connecter à une base de données Amazon Redshift.

Console

Connectez-vous à une base de données pour consulter des bases de données et des objets au sein de bases de données ou pour consulter des partages de données dans l'entrepôt de données Amazon Redshift. Les informations d'identification utilisateur utilisées pour se connecter

à une base de données spécifiée doivent disposer des autorisations nécessaires pour afficher toutes les unités de partage des données.

S’il n’y a pas de connexion locale, effectuez l’une des actions suivantes :

- Si vous êtes un administrateur producteur, accédez à l'onglet Clusters pour les clusters provisionnés ou à l'onglet Configuration des espaces de noms pour les points de terminaison sans serveur. Sélectionnez le cluster ou l'espace de noms correspondant dans la liste.
- Sur la page de détails du cluster ou de l'espace de noms, dans l'onglet Datashares, choisissez Connect to database et effectuez l'une des opérations suivantes :
  - Dans la section Partage de données provenant d'autres espaces de noms et Comptes AWS sections, consultez les partages de données provenant d'autres clusters, espaces de noms ou comptes.
  - Dans la section Partage de données créé dans mon cluster, consultez les partages de données de votre cluster.
- Dans la fenêtre Connect to database (Connexion à la base de données), procédez de l’une des manières suivantes :
  - Si vous choisissez Create a new connection (Créer une connexion), sélectionnez AWS Secrets Manager pour utiliser un secret stocké afin d’authentifier l’accès à la connexion.

    Sinon, sélectionnez Temporary credentials (Informations d’identification temporaires) pour utiliser les informations d’identification de la base de données afin d’authentifier l’accès à la connexion. Indiquez des valeurs pour Database name (Nom de base de données) et Database user (Utilisateur de la base de données).

    Choisissez Se connecter.
  - Choisissez Use a recent connection (Utiliser une connexion récente) pour vous connecter à une autre base de données pour laquelle vous disposez des autorisations nécessaires.

    Amazon Redshift établit automatiquement la connexion.

Une fois la connexion à la base de données établie, vous pouvez commencer à créer des partages de données, à interroger des partages de données ou à créer des bases de données à partir de partages de données.

Actions du producteur pour les nouveaux partages de données dans Amazon Redshift

Avec Amazon Redshift, vous pouvez partager des données en temps réel entre des clusters AWS ou des comptes Amazon Redshift à l'aide de partages de données. Un partage de données est un objet consommateur-producteur qui vous permet de partager des données en direct depuis votre cluster Amazon Redshift avec d'autres clusters ou comptes. AWS La création de partages de données permet un partage sécurisé des données tout en maintenant le contrôle de l'accès et en garantissant la conservation des données. up-to-date Les sections suivantes fournissent des informations détaillées sur la création de partages de données et l'ajout d'objets de base de données tels que des schémas, des tables et des vues pour partager des données en direct en toute sécurité.

Rubriques

- Création d'un partage de données dans Amazon Redshift
- Ajouter des objets à un partage de données dans Amazon Redshift
- Ajouter des consommateurs de données à un partage de données dans Amazon Redshift
- Autoriser un partage de données dans Amazon Redshift

Création d'un partage de données dans Amazon Redshift

Une unité de partage des données est un conteneur logique d’objets de base de données, d’autorisations et de consommateurs. Les consommateurs sont des clusters provisionnés par Amazon Redshift ou des espaces de noms Amazon Redshift Serverless dans votre compte, entre autres. Comptes AWS Chaque unité de partage des données est associée à la base de données dans laquelle elle a été créée, et seuls les objets de cette base de données peuvent être ajoutés. En tant qu'administrateur producteur, vous pouvez créer des partages de données sur la console et avec SQL en suivant l'une des procédures ci-dessous.

Console

Sur la console, vous pouvez créer des partages de données à partir des onglets Datashares de la page de détails du cluster ou de l'espace de noms. Une fois le partage de données créé, vous pouvez créer des bases de données à partir du partage de données sur un consommateur en tant qu'administrateur du consommateur.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/

2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Sur la page de détails du cluster ou de l'espace de noms, depuis l'onglet Partage de données, dans la section Partage de données, connectez-vous à une base de données si vous n'êtes pas connecté à une base de données. Dans la section Datashare created in my account, sélectionnez Create datashare. La page Créer une unité de partage des données s’affiche.
4. Choisissez Créer une unité de partage des données. Vous pouvez uniquement créer une unité de partage des données à partir d’une base de données locale. Si vous ne vous êtes pas connecté à la base de données au préalable, la page Se connecter à la base de données s’affiche. Suivez les étapes de Connexion à une base de données pour vous connecter à une base de données. S’il existe une connexion récente, la page Créer une unité de partage des données s’affiche.
5. Dans la section Informations sur l’unité de partage des données, choisissez l’une des options suivantes :
   - Choisissez Datashare pour créer des partages de données afin de partager des données à des fins de lecture ou d'écriture entre différents entrepôts de données Amazon Redshift (clusters provisionnés ou points de terminaison sans serveur) ou dans un environnement identique ou différent. Compte AWS Comptes AWS
   - Choisissez le partage AWS Data Exchange de données pour créer des partages de données via lesquels octroyer des licences à vos données. AWS Data Exchange
6. Spécifiez les valeurs pour Nom de l’unité de partage des données, Nom de la base de données et Accessible publiquement. Lorsque vous modifiez le nom de la base de données, créez une nouvelle connexion à la base de données.
7. Ajoutez des objets à votre partage de données à l'aide des sections Autorisations étendues ou Autorisations directes. Pour ajouter des objets à un partage de données, consultez. Création d'un partage de données dans Amazon Redshift
8. Dans la section Consommateurs de données, vous pouvez choisir de publier sur Amazon Redshift ou de publier sur le AWS Glue Data Catalog, ce qui lance le processus de partage de données avec Lake Formation. Publier votre partage de données sur Amazon Redshift signifie partager vos données avec un autre espace de noms ou un autre compte Amazon Redshift qui agit en tant que consommateur.

> **Note**
>
> Une fois l'unité de partage des données créée, vous ne pouvez pas modifier la configuration pour publier dans l'autre option.

9. Choisissez Créer une unité de partage des données.

SQL

La commande suivante crée une unité de partage des données :

```
CREATE DATASHARE salesshare;
```

Au moment de la création du partage de données, chaque partage de données est associé à une base de données. Seuls les objets de cette base de données peuvent être partagés dans ce datashare. Plusieurs datashares peuvent être créés sur la même base de données avec la même précision d'objets ou une précision différente. Il n'y a pas de limite au nombre d'unités de partage des données qu'un cluster peut créer. Vous pouvez également créer des datashares via la console Amazon Redshift. Pour de plus amples informations, veuillez consulter CREATE DATASHARE.

Vous pouvez également contrôler les restrictions de sécurité du partage de données lors de sa création. L'exemple suivant montre que le consommateur disposant d'un accès IP public est autorisé à lire le partage de données.

```
CREATE DATASHARE my_datashare [PUBLICACCESSIBLE = TRUE];
```

La définition de PUBLICACCESSIBLE = TRUE permet aux consommateurs d'interroger votre unité de partage des données à partir de clusters accessibles au public et de groupes de travail provisionnés. Oubliez ce paramètre ou définissez-le explicitement sur false si vous ne souhaitez pas l'autoriser.

Vous pouvez modifier les propriétés relatives au type de consommateurs après avoir créé un partage de données. Par exemple, vous pouvez définir que les clusters qui souhaitent consommer des données d'une unité de partage des données donné ne peuvent pas être accessibles publiquement. Les requêtes provenant de clusters consommateur qui ne respectent pas les restrictions de sécurité spécifiées dans l'unité de partage des données sont rejetées au

moment de l’exécution de la requête. Pour de plus amples informations, veuillez consulter ALTER DATASHARE.

Ajouter des objets à un partage de données dans Amazon Redshift

Vous pouvez ajouter des objets de base de données de différents types sur la console et avec SQL en suivant l'une des procédures ci-dessous.

Console

Vous pouvez ajouter des objets à votre partage de données à l'aide des sections Autorisations étendues ou Autorisations directes. Sélectionnez Accorder des autorisations étendues ou Accorder des autorisations directes pour ajouter des objets. Sélectionnez le bouton Ajouter pour ajouter des objets. Une boîte de dialogue s’affiche. Procédez comme suit :

1. Si vous sélectionnez Accorder des autorisations étendues, la page Autorisations étendues d'octroi apparaît, dans laquelle vous pouvez accorder des autorisations étendues au niveau d'une base de données ou d'un schéma. Les partages de données dotés d'autorisations étendues disposent des autorisations spécifiées sur tous les objets actuels et futurs de la base de données ou du schéma. Pour plus de détails, voir,Autorisations étendues.

    1. Ensuite, sélectionnez Autorisations étendues de base de données pour accorder des autorisations étendues au niveau de la base de données. Lorsque vous accordez des autorisations étendues, elles s'appliquent à la base de données actuelle lors de la création du partage de données. Ces autorisations ne peuvent pas être accordées à des objets individuels et s'appliquent à la fois aux objets existants et aux nouveaux objets (schémas, tables, vues UDFs).
    2. Sélectionnez les autorisations étendues pour les schémas, les tables, les vues ou les fonctions définies par l'utilisateur. Cela signifie que tous les objets de la base de données disposent des autorisations sélectionnées accordées aux consommateurs. Sélectionnez Accorder pour terminer l'octroi des autorisations étendues à la base de données.
    3. Sélectionnez ensuite les autorisations étendues du schéma pour accorder des autorisations étendues au niveau du schéma. Lorsque vous accordez des autorisations étendues au schéma, tous les objets ajoutés au schéma disposent des autorisations de partage de données spécifiées.

4. Sélectionnez le schéma que vous souhaitez ajouter au partage de données dans la liste déroulante. Vous ne pouvez sélectionner qu'un seul schéma à la fois. Sélectionnez ensuite les autorisations directes que vous souhaitez accorder sur le schéma sélectionné.
5. Sélectionnez des autorisations étendues pour les objets de votre schéma tels que les tables, les vues et les fonctions définies par l'utilisateur. Les autorisations sont accordées sur tous les objets correspondants du schéma. Il peut s'agir d'objets existants ou d'objets ajoutés dans le futur. Lorsqu'elle est appliquée, vous ne pouvez pas supprimer une autorisation d'un objet sans révoquer les autorisations délimitées.
6. Sélectionnez Accorder pour terminer l'octroi des autorisations délimitées par le schéma.

2. Si vous sélectionnez Accorder des autorisations directes, la page Accorder des autorisations directes apparaît. Vous pouvez y accorder des autorisations directes au niveau de chaque objet, tel qu'un schéma, une table, une vue ou une fonction définie par l'utilisateur. Pour accorder des autorisations directes, vous devez d'abord ajouter les schémas appropriés au partage de données.

   1. Ensuite, sélectionnez Accorder des autorisations directes aux schémas pour appliquer des autorisations directes à un schéma spécifique. Sélectionnez ensuite les autorisations de schéma pour vos objets de schéma tels que les tables, les vues et les fonctions définies par l'utilisateur, puis sélectionnez le schéma que vous souhaitez ajouter au partage de données. Sélectionnez Grant pour terminer l'ajout du schéma au partage de données.
   2. Une fois qu'un schéma a été ajouté à votre partage de données, vous pouvez ajouter des autorisations directes pour vos objets de schéma. Sélectionnez à nouveau Accorder des autorisations directes. La page Accorder des autorisations directes apparaît. Accédez ensuite aux onglets d'autorisations directes pour les objets du schéma.
   3. Sélectionnez Accorder des autorisations directes aux tables et aux vues pour accorder des autorisations directes au niveau des objets sur ces objets. Sélectionnez les autorisations directes requises et les objets requis dans la liste. Utilisez le champ de recherche pour trouver des objets de partage de données. Sélectionnez Grant pour terminer l'ajout de tables et de vues au partage de données.
   4. Sélectionnez Accorder des autorisations directes aux fonctions définies par l'utilisateur pour accorder des autorisations directes au niveau de l'objet sur les fonctions définies par l'utilisateur. Sélectionnez les autorisations directes requises et les objets requis dans la liste. Utilisez le champ de recherche pour trouver des objets de partage de données. Sélectionnez Grant pour terminer l'ajout de fonctions définies par l'utilisateur au partage de données.

3. Vous pouvez également choisir d'ajouter de futurs objets ou non. Lorsque vous choisissez d'inclure les objets d'unité de partage des données ajoutés au schéma, cela signifie que les objets ajoutés au schéma sont automatiquement ajoutés à l'unité de partage des données.
4. Choisissez Ajouter pour terminer la section et ajouter les objets. Ceux-ci sont répertoriés sous Objets d'unité de partage des données.
5. Après avoir ajouté des objets, vous pouvez sélectionner des objets individuels et modifier leurs autorisations. Si vous sélectionnez un schéma, une boîte de dialogue apparaît pour vous demander si vous souhaitez ajouter des autorisations étendues. Ainsi, chaque objet existant ou ajouté au schéma dispose d'un ensemble d'autorisations présélectionnées, adaptées au type d'objet. Par exemple, l'administrateur peut définir que toutes les tables ajoutées disposent des autorisations SELECT et UPDATE.
6. Tous les objets de partage de données sont répertoriés dans les sections Autorisations étendues ou Autorisations directes.
7. Dans la section Consommateurs de données, vous pouvez ajouter des espaces de noms ou ajouter AWS des comptes en tant que consommateurs du partage de données.
8. Sélectionnez Créer une unité de partage des données pour enregistrer vos modifications.

Une fois que vous avez créé l'unité de partage des données, il apparaît dans la liste sous Unités de partage des données créées dans mon espace de noms. Si vous choisissez une unité de partage des données dans la liste, vous pouvez afficher ses consommateurs, ses objets et d'autres propriétés.

SQL

Avec SQL, le propriétaire du partage de données doit accorder l'autorisation USAGE aux schémas qu'il souhaite ajouter au partage de données. Le GRANT est utilisé pour accorder diverses actions sur le schéma, notamment CREATE et USAGE. Les schémas contiennent des objets partagés :

```
CREATE SCHEMA myshared_schema1;
CREATE SCHEMA myshared_schema2;

GRANT USAGE ON SCHEMA myshared_schema1 TO DATASHARE my_datashare;
GRANT CREATE, USAGE ON SCHEMA myshared_schema2 TO DATASHARE my_datashare;
```

L'administrateur peut également continuer d'exécuter les commandes ALTER pour ajouter un schéma à l'unité de partage des données. Seules les autorisations USAGE sont accordées lorsqu'un schéma est ajouté de cette façon.

```
ALTER DATASHARE my_datashare ADD SCHEMA myshared_schema1;
```

Une fois que l'administrateur a ajouté des schémas, il peut accorder des autorisations d'unité de partage des données sur les objets du schéma. Il peut s'agit d'autorisations en lecture ou en écriture. L'exemple GRANT ALL montre comment accorder toutes les autorisations.

```
GRANT SELECT, INSERT ON TABLE myshared_schema1.table1, myshared_schema1.table2,
 myshared_schema2.table1
TO DATASHARE my_datashare;

GRANT ALL ON TABLE myshared_schema1.table4 TO DATASHARE my_datashare;
```

Vous pouvez continuer d'exécuter des commandes comme ALTER DATASHARE pour ajouter des tables. Dans ce cas, seules les autorisations SELECT sont accordées aux objets ajoutés.

```
ALTER DATASHARE my_datashare ADD TABLE myshared_schema1.table1,
 myshared_schema1.table2, myshared_schema2.table1;
```

Vous pouvez accorder des autorisations étendues à un partage de données sur tous les objets d'un type donné au sein d'une base de données ou d'un schéma. Les partages de données dotés d'autorisations étendues disposent des autorisations spécifiées sur tous les objets actuels et futurs de la base de données ou du schéma.

Vous pouvez consulter l'étendue des autorisations délimitées au niveau de la base de données dans. SVV_DATABASE_PRIVILEGES Vous pouvez consulter l'étendue des autorisations délimitées au niveau du schéma dans. SVV_SCHEMA_PRIVILEGES

Voici la syntaxe permettant d'accorder des autorisations étendues aux partages de données. Pour plus d'informations sur les autorisations délimitées, consultezAutorisations étendues.

```
GRANT { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }FOR SCHEMAS IN
DATABASE db_name
TO DATASHARE { datashare_name}

GRANT { { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [PRIVILEGES] } }FOR TABLES IN
```

```
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO DATASHARE { datashare_name}

GRANT { EXECUTE | ALL [ PRIVILEGES ] }FOR FUNCTIONS IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO DATASHARE { datashare_name}
```

Ajouter des consommateurs de données à un partage de données dans Amazon Redshift

Vous pouvez ajouter un ou plusieurs consommateurs de données aux partages de données sur la console ou avec SQL. Les consommateurs de données peuvent être des espaces de noms identifiant de manière unique les clusters Amazon Redshift ou. Comptes AWS

Console

Vous devez explicitement choisir de désactiver ou d'activer le partage de votre unité de partage des données vers des clusters avec accès public.

- Choisissez Ajouter des espaces de noms au partage de données. Les espaces de noms sont un identifiant global unique (GUID) pour le cluster Amazon Redshift.
- Choisissez Add Comptes AWS(Ajouter un Comptes AWS) pour l'unité de partage des données. La personne spécifiée Comptes AWS doit disposer d'autorisations d'accès au partage de données.

SQL

Avec SQL, l'administrateur autorise l'utilisation du partage de données à un espace de noms spécifique du compte. Vous pouvez trouver l'ID de l'espace de noms dans l'ARN disponible sur la page de détails du cluster, sur la page de détails de l'espace de noms Amazon Redshift sans serveur ou en exécutant la commande `SELECT current_namespace;`. Pour plus d'informations, consultez CURRENT_NAMESPACE.

```
GRANT USAGE ON DATASHARE my_datashare TO NAMESPACE '86b5169f-012a-234b-9fbb-
e2e24359e9a8';
```

Voici un exemple de la manière d'autoriser l'utilisation d'un partage de données à un. Compte AWS

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '123456789012';
```

L'exemple suivant montre comment autoriser l'utilisation d'une unité de partage des données à un compte Lake Formation.

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '123456789012' VIA DATA CATALOG;
```

Autoriser un partage de données dans Amazon Redshift

Avec Amazon Redshift, vous pouvez contrôler l'accès aux partages de données en autorisant certains consommateurs. Les partages de données vous permettent de partager des données en direct entre des clusters Amazon Redshift appartenant à un même compte ou à des comptes AWS différents, ce qui vous permet de distribuer et de consommer des données analytiques en toute simplicité. Cette section fournit des step-by-step instructions pour autoriser et révoquer l'accès des consommateurs à vos partages de données dans Amazon Redshift.

> **Note**
>
> Si vous ajoutez un espace de noms en tant que consommateur de données, vous n'avez pas à effectuer d'autorisation. Pour autoriser un partage de données, au moins un consommateur de données doit être ajouté au partage de données.

Console

En tant qu'administrateur producteur sur la console, vous pouvez choisir les consommateurs de données à autoriser à accéder aux partages de données ou auxquels vous souhaitez retirer leur autorisation. Les consommateurs de données autorisés reçoivent des notifications pour prendre des mesures sur les unités de partage des données. Si vous ajoutez un espace de noms en tant que consommateur de données, vous n'avez pas à effectuer d'autorisation.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Datashares. À partir de là, vous pouvez consulter une liste intitulée Consommateurs des unités de partage des données. Choisissez un ou plusieurs clusters consommateur que vous souhaitez autoriser. Choisissez ensuite Authorize (Autoriser).

3. La boîte de dialogue Autoriser un compte apparaît. Vous pouvez choisir parmi plusieurs types d'autorisation.

   - Lecture seule sur [nom du cluster ou du groupe de travail] : cela signifie qu'aucune autorisation d'écriture n'est disponible pour le consommateur, même si le créateur l'unité de partage des données a accordé des autorisations d'écriture.
   - Lecture et écriture sur [nom du cluster ou du groupe de travail] : cela signifie que toutes les autorisations attribuées par le créateur, y compris en écriture, sont disponible pour le consommateur.

4. Choisissez Enregistrer.

Vous pouvez également obtenir une autorisation AWS Data Exchange en tant que consommateur.

1. Si vous avez choisi Publier sur AWS Glue Data Catalog au moment de créer l'unité de partage des données, vous ne pouvez accorder l'autorisation de l'unité de partage des données qu'à un compte Lake Formation.

   Pour le AWS Data Exchange partage de données, vous ne pouvez autoriser qu'un seul partage de données à la fois.

   Lorsque vous autorisez un partage de AWS Data Exchange données, vous partagez le partage de données avec le AWS Data Exchange service et vous autorisez AWS Data Exchange à gérer l'accès au partage de données en votre nom. AWS Data Exchange permet l'accès aux consommateurs en ajoutant des comptes consommateurs en tant que consommateurs de données au AWS Data Exchange partage de données lorsqu'ils s'abonnent aux produits. AWS Data Exchange n'a pas d'accès en lecture au partage de données.

2. Choisissez Enregistrer.

Une fois que les consommateurs de données sont autorisés, ils peuvent accéder aux objets d'unité de partage des données et créer une base de données consommateur pour interroger les données.

API

L'administrateur de sécurité producteur détermine les éléments suivants :

- Si oui ou non un autre compte peut accéder à l'unité de partage des données.
- Si un compte dispose ou non d'autorisations d'écriture en cas d'accès à l'unité de partage des données.

Les autorisations IAM suivantes sont obligatoires pour autoriser une unité de partage des données :

décalage vers le rouge : AuthorizeDataShare

Vous pouvez autoriser l'utilisation et l'écriture à l'aide d'un appel CLI ou de l'API :

```
authorize-data-share
--data-share-arn <value>
--consumer-identifier <value>
[--allow-writes | --no-allow-writes]
```

Pour plus d'informations sur la commande, consultez authorize-data-share.

L'identifiant du consommateur peut être soit :

- Un identifiant de AWS compte à douze chiffres.
- L'ARN d'identifiant de l'espace de noms

> **Note**
>
> Les autorisations d'écriture ne sont pas accordées à l'étape d'autorisation. Autoriser l'écriture sur une unité de partage des données permet simplement au compte de disposer des autorisations d'écriture accordées par l'administrateur de l'unité de partage des données. Si un administrateur n'autorise pas les écritures, les seules autorisations disponibles pour le consommateur concerné sont SELECT, USAGE et EXECUTE.

Vous pouvez modifier l'autorisation d'un consommateur d'unité de partage des données en appelant `authorize-data-share` à nouveau, mais avec une valeur différente. L'ancienne autorisation est remplacée par la nouvelle. Ainsi, si vous autorisez et autorisez les écritures à l'origine, mais que vous réautorisez et spécifiez `no-allow-writes` ou simplement pas de valeur, le consommateur verra ses autorisations d'écriture révoquées.

Actions des consommateurs en faveur des nouveaux partages de données dans Amazon Redshift

Avec Amazon Redshift, vous pouvez utiliser des partages de données provenant d'autres AWS comptes, ce qui permet le partage de données et la collaboration entre comptes. Un partage de données est un moyen sécurisé de partager des données en temps réel entre des clusters Amazon Redshift, même s'ils se trouvent dans des comptes différents. AWS Les sections suivantes fournissent des étapes détaillées pour configurer l'accès, créer des bases de données à partir de partages de données, accorder des autorisations au niveau des objets et interroger des données partagées.

Rubriques

- Associer un partage de données à un autre Compte AWS dans Amazon Redshift
- Création d'une base de données à partir d'un partage de données dans Amazon Redshift
- Octroi d'autorisations au niveau de l'objet aux utilisateurs et aux rôles des consommateurs dans Amazon Redshift
- Interrogation de données dans un partage de données dans Amazon Redshift

Associer un partage de données à un autre Compte AWS dans Amazon Redshift

Avec Amazon Redshift, vous pouvez associer des partages de données partagés par d'autres Comptes AWS, ce qui permet un partage de données fluide et sécurisé au-delà des limites de l'organisation. Les partages de données sont des objets de base de données partageables qui encapsulent les données d'une ou de plusieurs bases de données Amazon Redshift. Les sections suivantes illustrent le processus d'association de partages de données.

Console

En tant qu'administrateur client, vous pouvez associer un ou plusieurs partages de données partagés depuis d'autres comptes à l'ensemble de votre AWS compte ou à des espaces de noms spécifiques de votre compte.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Datashares. La page de liste d’unités de partage des données s’affiche. Sélectionnez À partir d’autres comptes.

3. Dans la section Datashares from other accounts (Unités de partage des données provenant d'autres comptes), choisissez l'unité de partage des données que vous souhaitez associer, puis choisissez Associate (Associer). Lorsque la page Associer une unité de partage des données apparaît, choisissez l’un des types d’association suivants :

   - Choisissez AWS Compte entier pour associer au partage de données tous les espaces de noms existants et futurs AWS des différentes régions de votre AWS compte.
   - Si le partage de données est publié sur le AWS Glue Data Catalog, vous ne pouvez associer le partage de données qu'à l'ensemble du compte. AWS
4. À partir de là, vous pouvez choisir Autorisations acceptées. Les choix sont les suivants :

   - Lecture seule : si vous choisissez cette option, les autorisations d’écriture comme UPDATE ou INSERT ne sont pas disponibles pour le consommateur, même si ces autorisations ont été accordées et autorisées par le producteur.
   - Lecture et écriture : les utilisateurs de l’unité de partage des données consommateur disposeront de toutes les autorisations, à la fois en lecture et en écriture, accordées et autorisées par le producteur.
5. Vous pouvez également sélectionner AWS Régions et espaces de noms spécifiques pour associer une ou plusieurs AWS régions et espaces de noms spécifiques au partage de données. Choisissez Ajouter une région pour ajouter des AWS régions et des espaces de noms spécifiques au partage de données. La page Ajouter une AWS région apparaît.
6. Choisissez une région AWS .
7. Effectuez l’une des actions suivantes :

   - Choisissez Ajouter tous les espaces de noms pour ajouter tous les espaces de noms existants et futurs de cette région au partage de données.
   - Choisissez Ajouter des espaces de noms spécifiques pour ajouter au partage de données un ou plusieurs espaces de noms spécifiques de cette région.
   - Choisissez un ou plusieurs espaces de noms, puis choisissez Ajouter une AWS région.
8. Choisissez Associer.

Il est possible pour le producteur de revenir en arrière et de modifier les paramètres d’une autorisation, ce qui peut affecter les paramètres d’association des consommateurs.

Si vous associez l’unité de partage des données à un compte Lake Formation, accédez à la console Lake Formation pour créer une base de données, puis définissez les autorisations sur la

base de données. Pour plus d'informations, consultez la section Configuration des autorisations pour les partages de données Amazon Redshift dans le manuel du développeur. AWS Lake Formation Une fois que vous avez créé une AWS Glue base de données ou une base de données fédérée, vous pouvez utiliser l'éditeur de requêtes v2 ou n'importe quel client SQL préféré avec votre cluster client pour interroger les données.

Une fois l'unité de partage des données associée, les unités de partage des données deviennent disponibles.

> **Note**
>
> Vous pouvez également modifier l'association d'unité de partage des données à tout moment. Lorsque vous modifiez l'association de AWS régions et d'espaces de noms spécifiques à l'ensemble du AWS compte, Amazon Redshift remplace les informations relatives à la région et aux espaces de noms spécifiques par les informations du compte. AWS Toutes les AWS régions et tous les espaces de noms qu'il contient ont Compte AWS alors accès au partage de données.

API

> **Note**
>
> Les étapes décrites dans cette section sont exécutées une fois que l'administrateur du producteur a autorisé des actions spécifiques sur les objets de base de données partagés et que, si le partage de données est partagé avec un autre compte, l'administrateur de sécurité du producteur autorise l'accès.

L'administrateur de sécurité producteur détermine les éléments suivants :

- Si tous les espaces de noms d'un compte, des régions spécifiques du compte ou espaces de noms spécifiques ont accès à l'unité de partage des données.
- Si les espaces de noms ont accès à l'unité de partage des données, qu'ils disposent ou non d'autorisations d'écriture.

L'administrateur de Consumer Security peut associer le partage de données à la commande suivante :

```
associate-data-share-consumer
--data-share-arn <value>
--consumer-identifier <value>
[--allow-writes | --no-allow-writes]
```

Pour plus d'informations sur la commande, consultez associate-data-share-consumer.

L'administrateur de la sécurité consommateur doit définir explicitement la valeur `allow-writes` à true lorsqu'il associe une unité de partage des données à un espace de noms, afin de permettre l'utilisation des commandes INSERT et UPDATE. Dans le cas contraire, les utilisateurs ne peuvent effectuer que des opérations de lecture, comme les privilèges SELECT, USAGE ou EXECUTE.

Vous pouvez modifier l'association d'un espace de noms pour une unité de partage des données en appelant `associate-data-share-consumer` à nouveau, avec une valeur différente. L'ancienne association est remplacée par la nouvelle. Ainsi, si à l'origine vous avez associé et défini `allow-writes`, puis que vous associez et spécifiez `no-allow-writes`, ou ne spécifiez simplement pas de valeur, le consommateur verra ses autorisations d'écriture révoquées.

Création d'une base de données à partir d'un partage de données dans Amazon Redshift

Avec Amazon Redshift, vous pouvez utiliser un partage de données pour créer une base de données, puis interroger des données entre des partages de données provenant de clusters de producteurs afin d'accéder en toute sécurité aux données en temps réel sans les copier ni les transférer. Les étapes suivantes décrivent en détail la configuration d'une base de données dans votre environnement Amazon Redshift.

Console

Avant de pouvoir interroger des données dans le partage de données, vous devez créer une base de données à partir d'un partage de données. Vous ne pouvez créer qu'une seule base de données à partir d'une unité de partage des données spécifié.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Choisissez Datashares. La liste d'unités de partage des données s'affiche.

4. Dans la section Datashares from other clusters (Datashares provenant d'autres clusters), choisissez Connect to database (Connexion à la base de données). Pour plus d'informations, consultez Connexion à une base de données.
5. Choisissez une unité de partage des données à partir duquel vous voulez créer des bases de données, puis choisissez Créer une base de données à partir d'une unité de partage des données. La page Créer une base de données à partir d'une unité de partage des données s'affiche.
6. Dans Database name (Nom de base de données), spécifiez un nom de base de données. Le nom de base de données doit comporter 1 à 64 caractères alphanumériques (minuscules uniquement) et il ne peut pas s'agir d'un mot réservé.
7. Choisissez Créer.

Une fois la base de données créée, vous pouvez y interroger des données ou effectuer des opérations d'écriture, si elles ont été accordées, autorisées et associées par l'administrateur consommateur.

API

Pour partager des données à des fins de lecture en tant qu'administrateur client, effectuez les étapes suivantes.

1. Répertorier les unités de partage des données mises à votre disposition et afficher leur contenu. Pour plus d'informations, consultez DESC DATASHARE et SHOW DATASHARES.

   L'exemple suivant montre comment afficher les informations des unités de partage des données entrants d'un espace de noms producteur spécifié. Lorsque vous exécutez DESC DATASHARE en tant qu'administrateur client, vous devez spécifier l'option NAMESPACE pour afficher les partages de données entrants.

```
DESC DATASHARE salesshare OF NAMESPACE '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';


 producer_account  |          producer_namespace          | share_type |
 share_name | object_type |           object_name           |   include_new
-------------------+--------------------------------------+------------
+------------+-------------+----------------------------------+-----------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_users_redshift    |
```

```
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_venue_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_category_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_date_redshift     |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_event_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_listing_redshift  |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | table       | public.tickit_sales_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | schema      | public                          |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    |
 salesshare | view        | public.sales_data_summary_view  |
```

Seuls les super-utilisateurs de cluster peuvent le faire. Vous pouvez également utiliser SVV_DATASHARES pour afficher les unités de partage des données et SVV_DATASHARE_OBJECTS pour afficher les objets dans l'unité de partage des données.

L'exemple suivant affiche les datashares entrants dans un cluster consommateur.

```
SHOW DATASHARES LIKE 'sales%';


 share_name | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace
------------+-------------+-----------------+-------------------+------------
+------------+---------------------+-----------+------------------
+--------------------------------------
 salesshare |             |                 |                   | INBOUND
    |            |         t           |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

2. En tant que super-utilisateur de base de données, vous pouvez créer des bases de données locales qui font référence aux unités de partage des données. Pour de plus amples informations, veuillez consulter CREATE DATABASE.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Si vous souhaitez un contrôle plus précis de l'accès aux objets de la base de données locale, utilisez la clause WITH PERMISSIONS quand vous créez la base de données. Cela vous permet d'accorder des autorisations de niveau objet pour les objets de la base de données à l'étape 4.

```
CREATE DATABASE sales_db WITH PERMISSIONS FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Vous pouvez voir les bases de données que vous avez créées à partir de l'unité de partage des données en interrogeant la vue SVV_REDSHIFT_DATABASES. Vous pouvez vous connecter directement à ces bases de données ou vous connecter à une base de données locale sur votre cluster client et effectuer une requête entre bases de données pour interroger les données des bases de données partagées.

> **Note**
>
> Vous ne pouvez pas créer une unité de partage des données au-dessus des objets de base de données créés à partir d'une unité de partage des données existant. Toutefois, vous pouvez copier les données dans une table distincte du cluster consommateur, effectuer le traitement nécessaire, puis partager les nouveaux objets créés.

Vous pouvez également utiliser la console Amazon Redshift pour créer des bases de données à partir des unités de partage des données. Pour de plus amples informations, veuillez consulter Créer des bases de données à partir d'unités de partage des données.

Octroi d'autorisations au niveau de l'objet aux utilisateurs et aux rôles des consommateurs dans Amazon Redshift

En tant qu'administrateur client, vous pouvez accorder des autorisations aux utilisateurs et aux rôles au niveau de l'objet en effectuant les étapes suivantes.

SQL

Si vous avez créé votre base de données sans la clause WITH PERMISSIONS, vous ne pouvez attribuer des autorisations sur l'ensemble de la base de données créée à partir de l'unité de partage des données qu'à vos utilisateurs et groupes.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Vous pouvez également créer des vues à liaison tardive sur les objets partagés et les utiliser pour attribuer des autorisations détaillées. Vous pouvez également envisager que les clusters producteur créent des unités de partage des données supplémentaires pour vous avec la précision requise.

Si vous avez créé votre base de données avec WITH PERMISSIONS, vous devez attribuer des autorisations au niveau des objets de la base de données partagée. Un utilisateur disposant uniquement de l'autorisation USAGE ne peut accéder à aucun objet d'une base de données créée avec la clause WITH PERMISSIONS tant qu'il n'a pas obtenu d'autorisations de niveau objet supplémentaires.

```
GRANT SELECT ON sales_db.public.tickit_sales_redshift to Bob;
```

Pour plus d'informations sur l'octroi d'autorisations pour les écritures dans plusieurs entrepôts, consultezGestion des autorisations pour un partage de données dans Amazon Redshift.

Interrogation de données dans un partage de données dans Amazon Redshift

Avec Amazon Redshift, vous pouvez interroger des données provenant de clusters de producteurs dans des partages de données afin d'accéder en toute sécurité aux données en temps réel sans les copier ni les transférer. La section suivante traite des requêtes relatives aux partages de données dans votre environnement Amazon Redshift.

Les utilisateurs et les rôles disposant d'autorisations sur les bases de données consommateur et les schémas sur les clusters consommateur peuvent explorer et naviguer entre les métadonnées de tous les objets partagés. Ils peuvent également explorer et naviguer entre les objets locaux dans un cluster consommateur. Pour ce faire, ils utilisent des pilotes JDBC ou ODBC ou des vues SVV_ALL et SVV_REDSHIFT.

Les clusters producteur peuvent avoir de nombreux schémas dans la base de données, les tables et les vues au sein de chaque schéma. Les utilisateurs du côté consommateur ne peuvent voir que le sous-ensemble d'objets mis à disposition via l'unité de partage des données. Ces utilisateurs ne peuvent pas voir l'intégralité des métadonnées du cluster producteur. Cette approche permet de fournir un contrôle détaillé de la sécurité des métadonnées avec le partage des données.

Vous continuez à vous connecter aux bases de données de cluster locales. Mais maintenant, vous pouvez également lire à partir des bases de données et des schémas qui sont créés à partir de l'unité de partage des données en utilisant la notation database.schema.table en trois parties. Vous pouvez effectuer des requêtes qui s'étendent sur toutes les bases de données qui sont visibles pour vous. Il peut s'agir de bases de données locales sur le cluster ou de bases de données créées à partir des unités de partage des données. Les clusters consommateur ne peuvent pas se connecter aux bases de données créées à partir des unités de partage des données.

Vous pouvez accéder aux données à l'aide d'une qualification complète. Pour plus d'informations, consultez Exemples de requêtes entre bases de données.

SQL

```
SELECT * FROM sales_db.public.tickit_sales_redshift ORDER BY 1,2 LIMIT 5;

 salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime
---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------
       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |    728.00 |
  109.20 | 2008-02-18 02:36:48
       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |     76.00 |
   11.40 | 2008-06-06 05:00:16
       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |    350.00 |
   52.50 | 2008-06-06 08:26:17
       4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |    175.00 |
   26.25 | 2008-06-09 08:38:52
       5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |    154.00 |
   23.10 | 2008-08-31 09:17:02
```

Vous ne pouvez utiliser les instructions SELECT que sur les objets partagés. Toutefois, vous pouvez créer des tables dans le cluster consommateur en interrogeant les données des objets partagés dans une base de données locale différente.

Actions du producteur pour les partages de données existants dans Amazon Redshift

Avec Amazon Redshift, vous pouvez gérer les partages de données existants afin de contrôler l'accès à vos données dans un cluster Amazon Redshift. Les sections suivantes fournissent des step-by-step conseils sur la modification des objets de partage de données, la gestion des autorisations de partage de données et la mise à jour des propriétés de partage de données afin de contrôler et d'auditer efficacement l'accès aux données dans votre environnement Amazon Redshift.

Rubriques

- Afficher un partage de données dans Amazon Redshift
- Modification des partages de données créés dans votre compte dans Amazon Redshift
- Supprimer l'autorisation d'un partage de données dans Amazon Redshift
- Supprimer des objets de partage de données d'un partage de données dans Amazon Redshift
- Supprimer les consommateurs de données d'un partage de données dans Amazon Redshift
- Supprimer un partage de données créé dans votre compte dans Amazon Redshift

Afficher un partage de données dans Amazon Redshift

Vous pouvez consulter les partages de données depuis la console ou avec SQL.

Console

Vous pouvez consulter les partages de données depuis l'onglet Datashares ou Clusters.

- Utilisez l'onglet Datashares pour répertorier les partages de données de votre compte ou d'autres comptes.
  - Pour afficher les unités de partage des données créées dans votre compte, choisissez Dans mon compte, puis choisissez l’unité de partage des données que vous souhaitez afficher.
  - Pour afficher les unités de partage des données partagées à partir d’autres comptes, choisissez Provenant d’autres comptes, puis choisissez l’unité de partage des données que vous souhaitez afficher.
- Utilisez l'onglet Clusters pour répertorier les partages de données de votre cluster ou d'autres clusters.

  Connectez-vous d’abord à une base de données. Choisissez ensuite un partage de données dans la section Datashares from other clusters ou Datashares created in my cluster pour en afficher les détails.

SQL

Vous pouvez répertorier les partages de données créés dans le cluster et consulter le contenu du partage de données.

L'exemple suivant montre comment afficher les informations d'une unité de partage des données nommée `salesshare`.

```
DESC DATASHARE salesshare;

 producer_account  |          producer_namespace          | share_type | share_name
 | object_type |           object_name          |   include_new
------------------+--------------------------------------+------------+------------
+-------------+--------------------------------+------------------
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_users_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_venue_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_category_redshift|
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_date_redshift    |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_event_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_listing_redshift |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | table       | public.tickit_sales_redshift   |
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | schema      | public                         |  t
 123456789012      | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | OUTBOUND   | salesshare
 | view        | public.sales_data_summary_view |
```

L'exemple suivant montre comment afficher les datashares sortants dans un cluster producteur.

```
SHOW DATASHARES LIKE 'sales%';
```

La sortie ressemble à ce qui suit.

```
share_name | share_owner  | source_database | consumer_database | share_type |
  createdate      | is_publicaccessible  | share_acl | producer_account |
 producer_namespace
```

```
----------+-------------+----------------+-------------------+-----------
+--------------------+---------------------+----------+-----------------
+--------------------------------------
salesshare |    100       | dev             |                   |  OUTBOUND
 | 2020-12-09 02:27:08 |          True        |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

Pour plus d'informations, consultez DESC DATASHARE et SHOW DATASHARES.

Vous pouvez également utiliser SVV_DATASHARES, SVV_DATASHARE_CONSUMERS, et SVV_DATASHARE_OBJECTS pour afficher les datashares, les objets du datashare et les consommateurs de datashare.

Modification des partages de données créés dans votre compte dans Amazon Redshift

Vous pouvez modifier un partage de données créé dans votre compte à l'aide de la console et de SQL.

Console

Sur la console, suivez les étapes ci-dessous pour vous connecter d'abord à une base de données afin de voir la liste des partages de données créés dans votre compte.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Choisissez Datashares.
4. Dans la section Datashares created in my account (Datashares créés sur mon compte), choisissez Connect to database (Connexion à la base de données).
5. Choisissez l'unité de partage des données à modifier, puis choisissez Modifier. La page de détails de l'unité de partage des données s'affiche.
6. Effectuez des modifications dans la section Datashare objects (Objets de datashare) ou Data consumers (Consommateurs de données).
7. Sélectionnez Enregistrer les modifications. Amazon Redshift met à jour votre unité de partage des données avec les modifications.

> **Note**
>
> Si vous avez choisi de publier votre partage de données sur le AWS Glue Data Catalog, vous ne pouvez pas modifier la configuration pour publier le partage de données sur d'autres comptes Amazon Redshift.

SQL

Utilisez ALTER DATASHARE pour supprimer des objets des datashares à n'importe quel point du datashare. Pour supprimer un schéma, utilisez la commande suivante :

```
ALTER DATASHARE salesshare REMOVE SCHEMA PUBLIC;
```

Pour supprimer une table, utilisez la commande suivante :

```
ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
```

Utilisez REVOKE USAGE ON pour révoquer les autorisations de certains consommateurs sur l’unité de partage des données. Il révoque les autorisations USAGE sur les objets dans une unité de partage des données et arrête instantanément l’accès à tous les clusters consommateur. Répertorier des unités de partage des données et des requêtes de métadonnées, telles que la liste des bases de données et des tables, ne renvoie pas les objets partagés après la révocation de l’accès. Révoquez l’accès à l’unité de partage des données depuis les espaces de noms si vous ne souhaitez plus partager les données avec les consommateurs.

```
REVOKE USAGE ON DATASHARE salesshare FROM NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Si vous ne souhaitez plus partager les données avec les consommateurs, révoquez l'accès au partage de données auprès de : Comptes AWS

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '123456789012';
```

Supprimer l'autorisation d'un partage de données dans Amazon Redshift

Avec Amazon Redshift, vous pouvez contrôler l'accès aux partages de données en révoquant l'autorisation de certains consommateurs. Cette section fournit des instructions pour révoquer l'accès des consommateurs à vos partages de données dans Amazon Redshift.

> **Note**
>
> Pour supprimer l'autorisation pour le partage de données, au moins un consommateur de données doit être ajouté au partage de données.

Console

Choisissez un ou plusieurs clusters consommateur pour lesquels vous souhaitez supprimer l’autorisation. Choisissez ensuite Supprimer l'autorisation.

Une fois l’autorisation supprimée, les consommateurs de données perdent immédiatement l’accès à l’unité de partage des données.

API

L’administrateur de sécurité producteur détermine les éléments suivants :

- Si oui ou non un autre compte peut accéder à l’unité de partage des données.
- Si un compte dispose ou non d’autorisations d’écriture en cas d’accès à l’unité de partage des données.

Les autorisations IAM suivantes sont requises pour annuler l'autorisation d'un partage de données :

décalage vers le rouge : DeauthorizeDataShare

Vous pouvez annuler l'autorisation d'utilisation et d'écriture à l'aide d'un appel CLI ou de l'API :

```
deauthorize-data-share
--data-share-arn <value>
--consumer-identifier <value>
```

Pour plus d'informations sur la commande, consultez deauthorize-data-share.

Supprimer des objets de partage de données d'un partage de données dans Amazon Redshift

Vous pouvez supprimer un ou plusieurs objets d’une unité de partage des données à l’aide de la procédure suivante.

Console

Pour supprimer un ou plusieurs objets d'un partage de données sur la console, procédez comme suit.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Choisissez Datashares.
4. Dans la section Datashares created in my account (Datashares créés sur mon compte), choisissez Connect to database (Connexion à la base de données). Pour plus d'informations, consultez Connexion à une base de données.
5. Choisissez l’unité de partage des données à modifier, puis choisissez Modifier. La page de détails de l’unité de partage des données s’affiche.
6. Pour supprimer un ou plusieurs objets d’unité de partage des données d’une unité de partage des données, effectuez l’une des actions suivantes :

   - Pour supprimer des schémas d’une unité de partage des données, sélectionnez un ou plusieurs schémas. Puis choisissez Remove (Supprimer). Amazon Redshift supprime les schémas spécifiés et tous les objets des schémas spécifiés de l’unité de partage des données.
   - Pour supprimer des tables et des vues de l’unité de partage des données, sélectionnez une ou plusieurs tables et vues. Puis choisissez Remove (Supprimer). Sinon, choisissez Remove by schema (Supprimer par schéma) pour supprimer toutes les tables et vues des schémas spécifiés.
   - Pour supprimer des fonctions définies par l’utilisateur de l’unité de partage des données, sélectionnez une ou plusieurs fonctions définies par l’utilisateur. Puis choisissez Remove (Supprimer). Sinon, choisissez Remove by schema (Supprimer par schéma) pour supprimer toutes les fonctions définies par l’utilisateur dans les schémas spécifiés.

SQL

Utilisez ALTER DATASHARE pour supprimer des objets des datashares à n'importe quel point du datashare. Pour supprimer un schéma, utilisez la commande suivante :

```
ALTER DATASHARE salesshare REMOVE SCHEMA PUBLIC;
```

Pour supprimer une table, utilisez la commande suivante :

```
ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
```

Supprimer les consommateurs de données d'un partage de données dans Amazon Redshift

Vous pouvez supprimer un ou plusieurs consommateurs de données d’une unité de partage des données. Les consommateurs de données peuvent être des espaces de noms qui identifient de manière unique les clusters AWS ou les comptes Amazon Redshift.

Console

Pour supprimer un ou plusieurs consommateurs de données d'un partage de données sur la console, choisissez-en un ou plusieurs dans l'espace de noms ou dans le compte IDs . AWS Choisissez ensuite Supprimer.

Amazon Redshift supprime les consommateurs de données spécifiés de l’unité de partage des données. Ils perdent immédiatement l’accès au partage de données.

SQL

Utilisez REVOKE USAGE ON pour révoquer les autorisations de certains consommateurs sur l’unité de partage des données. Il révoque les autorisations USAGE sur les objets dans une unité de partage des données et arrête instantanément l’accès à tous les clusters consommateur. Répertorier des unités de partage des données et des requêtes de métadonnées, telles que la liste des bases de données et des tables, ne renvoie pas les objets partagés après la révocation de l’accès. Révoquez l’accès à l’unité de partage des données depuis les espaces de noms si vous ne souhaitez plus partager les données avec les consommateurs.

```
REVOKE USAGE ON DATASHARE salesshare FROM NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Supprimer un partage de données créé dans votre compte dans Amazon Redshift

Vous pouvez supprimer un partage de données créé dans votre compte à l'aide de la console ou de SQL.

Console

Pour supprimer un partage de données créé dans votre compte à l'aide de la console, connectez-vous d'abord à une base de données pour voir la liste des partages de données créés dans votre compte.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez votre cluster. La page de détails du cluster s'affiche.
3. Choisissez Datashares. La liste d’unités de partage des données s’affiche.
4. Dans la section Datashares created in my account (Datashares créés sur mon compte), choisissez Connect to database (Connexion à la base de données).
5. Choisissez une ou plusieurs unités de partage des données à supprimer, puis sélectionnez Supprimer. La page Supprimer des unités de partage des données s’affiche.

   La suppression d’une unité de partage des données partagée avec Lake Formation ne supprime pas automatiquement les autorisations associées dans Lake Formation. Pour les supprimer, accédez à la console Lake Formation.
6. Saisissez Supprimer pour confirmer la suppression des unités de partage des données spécifiées.
7. Sélectionnez Delete (Supprimer).

Une fois les unités de partage des données supprimées, les consommateurs d’unités de partage des données perdent l’accès aux unités de partage des données.

SQL

Vous pouvez utiliser SQL pour supprimer les objets de partage de données à tout moment en utilisant. DROP DATASHARE Les super-utilisateurs de cluster et les propriétaires d’unité de partage des données peuvent supprimer des unités de partage des données.

L’exemple suivant supprime une unité de partage des données appelé `salesshare`.

```
DROP DATASHARE salesshare;
```

Actions des consommateurs concernant les partages de données existants dans Amazon Redshift

Avec Amazon Redshift, vous pouvez gérer les partages de données existants afin de contrôler l'accès à vos données dans un cluster Amazon Redshift. Les sections suivantes fournissent des step-by-step conseils sur la gestion des partages de données dans votre environnement Amazon Redshift en tant qu'administrateur client.

Rubriques

- Gestion des autorisations pour un partage de données dans Amazon Redshift
- Supprimer l'association d'un partage de données aux consommateurs de données dans un autre environnement dans Amazon Compte AWS Redshift
- Refuser un partage de données par rapport à un autre Compte AWS dans Amazon Redshift

Gestion des autorisations pour un partage de données dans Amazon Redshift

En tant qu'administrateur producteur, vous gardez le contrôle des ensembles de données que vous partagez. Vous pouvez ajouter de nouveaux objets à l'unité de partage des données ou en supprimer. Vous pouvez également accorder ou révoquer l'accès aux partages de données dans leur ensemble pour les clusters de consommateurs, les AWS comptes ou les régions. AWS Lorsque les autorisations sont révoquées, les clusters consommateur perdent immédiatement l'accès aux objets partagés et cessent de les voir dans la liste des unités de partage des données INBOUND dans SVV_DATASHARES.

L'exemple suivant crée le partage de données`salesshare`, ajoute le schéma `public` et ajoute la table `public.tickit_sales_redshift` à. `salesshare` Il accorde également des autorisations d'utilisation sur `salesshare` l'espace de noms spécifié.

```
CREATE DATASHARE salesshare;

ALTER DATASHARE salesshare ADD SCHEMA public;

ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
```

```
GRANT USAGE ON DATASHARE salesshare TO NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Pour CREATE DATASHARE, les super-utilisateurs et les propriétaires de bases de données peuvent créer des unités de partage des données. Pour plus d'informations, consultez CREATE DATASHARE. Pour ALTER DATASHARE, le propriétaire de l'unité de partage des données disposant des autorisations requises sur les objets d'unité de partage des données à ajouter ou à supprimer peut modifier l'unité de partage des données. Pour plus d'informations, veuillez consulter ALTER DATASHARE.

En tant qu'administrateur de producteurs, lorsque vous supprimez une unité de partage des données, il cesse d'être répertorié sur les clusters consommateur. Les bases de données et les références de schéma créées sur le cluster consommateur à partir de l'unité de partage des données supprimé continuent d'exister sans objet. L'administrateur du client doit supprimer ces bases de données manuellement.

Du côté du consommateur, un administrateur du consommateur peut déterminer quels utilisateurs et quels rôles doivent avoir accès aux données partagées en créant une base de données à partir du partage de données. En fonction des options que vous choisissez lors de la création de la base de données, vous pouvez contrôler l'accès à celle-ci comme suit. Pour plus d'informations sur la création d'une base de données à partir d'une unité de partage des données, consultez CREATE DATABASE.

Pour plus d'informations sur la configuration d'un partage de données et la lecture des données d'un consommateur, voir Partage de l'accès en lecture aux données d'un AWS compte.

Création de la base de données sans la clause WITH PERMISSIONS

Un administrateur peut contrôler l'accès au niveau de la base de données ou du schéma. Pour contrôler l'accès au niveau du schéma, l'administrateur doit créer un schéma externe à partir de la base de données Amazon Redshift créée à partir de l'unité de partage des données.

L'exemple suivant accorde des autorisations pour accéder à une table partagée au niveau de la base de données et du schéma.

```
GRANT USAGE ON DATABASE sales_db TO Bob;

CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE sales_db SCHEMA 'public';

GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

Pour restreindre davantage l’accès, vous pouvez créer des vues au-dessus des objets partagés, en exposant uniquement les données nécessaires. Vous pouvez ensuite utiliser ces vues pour donner accès aux utilisateurs et aux rôles.

Une fois que les utilisateurs ont obtenu l'accès à la base de données ou au schéma, ils auront accès à tous les objets partagés de cette base de données ou de ce schéma.

Création de la base de données avec la clause WITH PERMISSIONS

Après avoir accordé des droits d’utilisation sur la base de données ou le schéma, un administrateur peut contrôler davantage l’accès en utilisant le même processus d’octroi d’autorisations que sur une base de données ou un schéma local. Sans autorisations sur les objets individuels, les utilisateurs ne peuvent accéder à aucun objet de la base de données de l’unité de partage des données ou du schéma, même après avoir obtenu l’autorisation USAGE.

L’exemple suivant octroie des autorisations pour accéder à une table partagée au niveau de la base de données.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
GRANT USAGE FOR SCHEMAS IN DATABASE sales_db TO Bob;
GRANT SELECT ON sales_db.public.tickit_sales_redshift TO Bob;
```

Après avoir obtenu l’accès à la base de données ou au schéma, les utilisateurs doivent toujours disposer des autorisations appropriées pour tous les objets de la base de données ou du schéma auxquels vous souhaitez qu’ils accèdent.

Partage granulaire à l'aide de WITH PERMISSIONS

Vous pouvez utiliser le partage granulaire à l'aide de WITH PERMISSIONS pour permettre aux clusters ou aux groupes de travail sans serveur d'interroger le partage de données. Ce processus suppose que le partage de données provient d'un autre cluster ou d'un autre espace de noms Amazon Redshift Serverless de votre compte, ou qu'il provient d'un autre compte et a été associé à l'espace de noms que vous utilisez.

1. L’administrateur de la base de données consommateur peut créer une base de données à partir de l’unité de partage des données.

   ```
   CREATE DATABASE my_ds_db [WITH PERMISSIONS] FROM DATASHARE my_datashare OF
    NAMESPACE 'abc123def';
   ```

Si vous créez une base de données AVEC PERMISSIONS, vous pouvez accorder des autorisations granulaires sur les objets de partage de données à différents utilisateurs et rôles. Sans cela, tous les utilisateurs et rôles disposant de l'autorisation USAGE sur la base de données de l'unité de partage des données obtiennent toutes les autorisations sur tous les objets celle-ci.

2. Voici comment accorder des autorisations à un rôle ou à un utilisateur de base de données Redshift. Vous devez être connecté à une base de données locale pour exécuter ces instructions. Vous ne pouvez pas exécuter ces instructions si vous exécutez une commande USE sur la base de données de l'unité de partage des données avant d'exécuter les instructions GRANT.

```
GRANT USAGE ON DATABASE my_ds_db TO ROLE data_eng;
GRANT CREATE, USAGE ON SCHEMA my_ds_db.my_shared_schema TO ROLE data_eng;
GRANT ALL ON ALL TABLES IN SCHEMA my_ds_db.my_shared_schema TO ROLE data_eng;

GRANT USAGE ON DATABASE my_ds_db TO bi_user;
GRANT USAGE ON SCHEMA my_ds_db.my_shared_schema TO bi_user;
GRANT SELECT ON my_ds_db.my_shared_schema.table1 TO bi_user;
```

Supprimer l'association d'un partage de données aux consommateurs de données dans un autre environnement dans Amazon Compte AWS Redshift

Avec Amazon Redshift, vous pouvez supprimer l'association des partages de données partagés par d'autres utilisateurs. Comptes AWS Les partages de données sont des objets de base de données partageables qui encapsulent les données d'une ou de plusieurs bases de données Redshift. Les sections suivantes montrent le processus de dissociation des partages de données au sein de votre environnement Redshift.

Console

En tant qu'administrateur de consommateurs, vous pouvez supprimer l'association de partages de données aux consommateurs de données sur la console.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Datashares. La page de liste d'unités de partage des données s'affiche.

3. Sélectionnez À partir d’autres comptes.
4. Dans la section Datashares from other accounts (Datashares provenant d'autres comptes), choisissez le datashare duquel supprimer l'association des consommateurs de données.
5. Dans la section Data consumers (Consommateurs de données), sélectionnez un ou plusieurs consommateurs de données desquels supprimer l’association. Puis choisissez Remove association (Supprimer l’association).
6. Lorsque la page Remove association (Supprimer l’association) s’affiche, choisissez Remove association (Supprimer l’association).

Une fois l’association supprimée, les consommateurs de données perdront l’accès à l’unité de partage des données. Vous pouvez modifier l’association de consommateurs de données à tout moment.

SQL

> **Note**
>
> Les étapes décrites dans cette section sont exécutées une fois que l'administrateur du producteur a autorisé des actions spécifiques sur les objets de base de données partagés et que, si le partage de données est partagé avec un autre compte, l'administrateur de sécurité du producteur autorise l'accès.

L'administrateur de Consumer Security peut dissocier le partage de données à l'aide de la commande suivante :

```
disassociate-data-share-consumer
--data-share-arn <value>
```

Pour plus d'informations sur la commande, consultez disassociate-data-share-consumer.

Refuser un partage de données par rapport à un autre Compte AWS dans Amazon Redshift

Avec Amazon Redshift, vous pouvez refuser les partages de données partagés par d'autres AWS comptes, ce qui permet un partage de données fluide et sécurisé au-delà des frontières de l'organisation. Les partages de données sont des objets de base de données partageables qui encapsulent les données d'une ou de plusieurs bases de données Amazon Redshift.

En tant qu'administrateur client, vous pouvez rejeter tout partage de données dont l'état est disponible ou actif. Après avoir rejeté une unité de partage des données, les utilisateurs du cluster consommateur perdent l'accès à l'unité de partage des données. Amazon Redshift ne renvoie pas l'unité de partage des données rejetée si vous appelez l'opération d'API `DescribeDataSharesForConsumer`. Si l'administrateur du producteur exécute l'opération `DescribeDataSharesForProducer` API, il constatera que le partage de données a été rejeté. Lorsqu'un partage de données est rejeté, l'administrateur du producteur peut à nouveau autoriser le partage de données à un cluster de consommateurs, et l'administrateur du consommateur peut choisir d'associer son AWS compte au partage de données ou de le rejeter.

Si votre AWS compte est associé à un partage de données et qu'il existe une association en attente à un partage de données géré par Lake Formation, le rejet de l'association de partage de données gérée par Lake Formation entraîne également le rejet du partage de données d'origine. Pour rejeter une association spécifique, l'administrateur du producteur peut supprimer l'autorisation d'un partage de données spécifié. Cette action n'affecte pas les autres unités de partage des données.

Pour rejeter un partage de données, utilisez la AWS console, l'opération `RejectDataShare` API ou `reject-datashare` dans le. AWS CLI

Console

Pour rejeter un partage de données à l'aide de la AWS console, effectuez les opérations suivantes.

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, sélectionnez Unités de partage des données.
3. Sélectionnez À partir d'autres comptes.
4. Dans la section Datashares from other accounts (Datashares provenant d'autres comptes), choisissez le datashare que vous souhaitez refuser. Lorsque la page Refuser l'unité de partage des données s'affiche, choisissez Refuser.

Une fois que vous avez refusé les unités de partage des données, vous ne pouvez pas revenir en arrière. Amazon Redshift supprime les unités de partage des données de la liste. Pour voir à nouveau l'unité de partage des données, l'administrateur du producteur doit l'autoriser à nouveau.

CLI

Pour rejeter un partage de données, l'administrateur utilise la commande suivante :

```
reject-data-share
--data-share-arn <value>
```

Pour plus d'informations sur la commande, consultez reject-data-share.

Commencer à partager des données avec AWS CloudFormation Amazon Redshift

Vous pouvez automatiser la configuration du partage de données en utilisant une AWS CloudFormation pile qui fournit des AWS ressources. La CloudFormation pile configure le partage de données entre deux clusters Amazon Redshift d'un même AWS compte. Ainsi, vous pouvez commencer à partager des données sans exécuter d'instructions SQL pour mettre en service vos ressources.

La pile crée une unité de partage des données sur le cluster que vous désignez. L'unité de partage des données inclut une table et des exemples de données en lecture seule. Ces données peuvent être lues par votre autre cluster Amazon Redshift.

Si vous souhaitez commencer à partager des données dans un AWS compte en exécutant des instructions SQL pour configurer un partage de données et octroyer des autorisations, sans les utiliser CloudFormation, consultez. Partage de l'accès en lecture aux données au sein d'un Compte AWS

Avant d'exécuter la CloudFormation pile de partage de données, vous devez être connecté avec un utilisateur autorisé à créer un rôle IAM et une fonction Lambda. Vous avez également besoin de deux clusters Amazon Redshift dans le même compte. Vous en utilisez un, le producteur, pour partager les exemples de données, et l'autre, le consommateur, pour les lire. La principale exigence pour ces clusters est que chacun utilise RA3 des nœuds. Pour des prérequis supplémentaires, consultez Considérations relatives au partage de données dans Amazon Redshift.

Pour plus d'informations sur la mise en place d'un cluster Amazon Redshift, consultez Commencer avec les entrepôts de données provisionnés Amazon Redshift. Pour plus d'informations sur l'automatisation de la configuration avec CloudFormation, voir Qu'est-ce que c'est ? AWS CloudFormation

> **Important**
>
> Avant de lancer votre CloudFormation stack, assurez-vous que vous disposez de deux clusters Amazon Redshift dans le même compte et que les clusters utilisent RA3 des nœuds. Assurez-vous que chaque cluster possède une base de données et un super-utilisateur. Pour plus d'informations, consultez CREATE DATABASE et superuser.

Pour lancer votre CloudFormation stack pour le partage de données Amazon Redshift :

1. Cliquez sur Lancer la pile CFN, qui vous amène au CloudFormation service dans le AWS Management Console.

   Si vous y êtes invité, connectez-vous.

   Le processus de création de la pile démarre en faisant référence à un fichier CloudFormation modèle, qui est stocké dans Amazon S3. Un CloudFormation modèle est un fichier texte au format JSON qui déclare AWS les ressources qui constituent une pile. Pour plus d'informations sur les CloudFormation modèles, voir Apprendre les principes de base des modèles.
2. Choisissez Next (Suivant) pour saisir les détails de la pile.
3. Sous Parameters (Paramètres), pour chaque cluster, saisissez les éléments suivants :

   - Le nom de votre cluster Amazon Redshift, par exemple **`ra3-consumer-cluster`**
   - Le nom de votre base de données, par exemple **`dev`**
   - Le nom de l'utilisateur de votre base de données, par exemple **`consumeruser`**

   Nous vous recommandons d'utiliser des clusters de test, car la pile crée plusieurs objets de base de données.

   Choisissez Suivant.
4. Les options de pile apparaissent.

   Choisissez Next (Suivant) pour accepter les paramètres par défaut.
5. Sous Capabilities (Capacités), choisissez I acknowledge that AWS CloudFormation might create IAM resources (Je comprends qu' AWS CloudFormation peut créer des ressources IAM).
6. Sélectionnez Créer la pile.

CloudFormation il faut environ 10 minutes pour créer la pile Amazon Redshift à l'aide du modèle, en créant un partage de données appelé. `myproducer_share` La pile crée l'unité de partage des données dans la base de données spécifiée dans les détails de la pile. Seuls les objets de cette base de données peuvent être partagés.

Si une erreur se produit pendant la création de la pile, procédez comme suit :

- Assurez-vous d'avoir saisi le nom de cluster Amazon Redshift, le nom de base de données et le nom d'utilisateur de la base de données corrects pour chaque cluster Redshift.
- Assurez-vous que votre cluster possède RA3 des nœuds.
- Assurez-vous d'être connecté avec un utilisateur qui a l'autorisation de créer un rôle IAM et une fonction Lambda.. Pour plus d'informations sur la création de rôles IAM, consultez Création de rôles IAM. Pour plus d'informations sur les politiques de création de fonctions Λ, consultez Développement des fonctions.

Interrogation de l'unité de partage des données que vous avez créée

Pour utiliser la procédure suivante, vérifiez que vous disposez des autorisations requises pour exécuter des requêtes sur chaque cluster décrit.

Pour interroger votre unité de partage des données :

1. Connectez-vous au cluster de producteurs sur la base de données saisie lors de la création de votre CloudFormation stack, à l'aide d'un outil client tel que l'éditeur de requêtes Amazon Redshift v2.
2. Requête pour les unités de partage des données.

```
SHOW DATASHARES;

+------------------+-------------+-----------------+------------------
+------------+------------+--------------------+-----------+-----------------
+--------------------------------------+
|    share_name    | share_owner | source_database | consumer_database | share_type
 | createdate | is_publicaccessible | share_acl | producer_account |
 producer_namespace          |
+------------------+-------------+-----------------+------------------
+------------+------------+--------------------+-----------+-----------------
+--------------------------------------+
```

```
| myproducer_share | 100         | sample_data_dev | myconsumer_db     | INBOUND
  | NULL       | true               | NULL      |  producer-acct   |        your-
producer-namespace       |
+------------------+-------------+-----------------+-------------------
+------------+------------+--------------------+-----------+-----------------
+-------------------------------------+
```

La commande précédente renvoie le nom de l'unité de partage des données créée par la pile, appelée `myproducer_share`. Elle renvoie également le nom de base de données associé à l'unité de partage des données `myconsumer_db`.

Copiez l'identifiant de l'espace de noms du producteur à utiliser lors d'une étape ultérieure.

3. Décrivez les objets dans l'unité de partage des données.

```
DESC DATASHARE myproducer_share;

+------------------+---------------------------------------+------------
+------------------+-------------+--------------------------------------
+-------------+
| producer_account |          producer_namespace          | share_type |
 share_name    | object_type |             object_name             | include_new |
+------------------+---------------------------------------+------------
+------------------+-------------+--------------------------------------
+-------------+
|   producer-acct  |        your-producer-namespace       | OUTBOUND   |
 myproducer_share | schema      | myproducer_schema                    | true
 |
|   producer-acct  |        your-producer-namespace       | OUTBOUND   |
 myproducer_share | table       | myproducer_schema.tickit_sales       | NULL
 |
|   producer-acct  |        your-producer-namespace       | OUTBOUND   |
 myproducer_share | view        | myproducer_schema.ticket_sales_view | NULL
 |
+------------------+---------------------------------------+------------
+------------------+-------------+--------------------------------------
+-------------+
```

Lorsque vous décrivez l'unité de partage des données, il renvoie les propriétés des tables et des vues. La pile ajoute des tables et des vues contenant des exemples de données à la base de données producteur, par exemple `tickit_sales` et `tickit_sales_view`. Pour

plus d'informations sur l'exemple de base de données TICKIT, consultez Exemple de base de données.

Il n'est pas nécessaire de déléguer des autorisations sur l'unité de partage des données pour exécuter des requêtes. La pile accorde les autorisations nécessaires.

4. Connectez-vous au cluster consommateur à l'aide de votre outil client. Décrivez l'unité de partage des données, en spécifiant l'espace de noms du producteur.

```
DESC DATASHARE myproducer_share OF NAMESPACE '<namespace id>'; --specify the unique
 identifier for the producer namespace

+------------------+--------------------------------------+------------
+------------------+-------------+-------------------------------------
+-------------+
| producer_account |          producer_namespace          | share_type |
 share_name    | object_type |             object_name             | include_new |
+------------------+--------------------------------------+------------
+------------------+-------------+-------------------------------------
+-------------+
|   producer-acct  |        your-producer-namespace       | INBOUND    |
 myproducer_share | schema      | myproducer_schema                   | NULL
 |
|   producer-acct  |        your-producer-namespace       | INBOUND    |
 myproducer_share | table       | myproducer_schema.tickit_sales      | NULL
 |
|   producer-acct  |        your-producer-namespace       | INBOUND    |
 myproducer_share | view        | myproducer_schema.ticket_sales_view | NULL
 |
+------------------+--------------------------------------+------------
+------------------+-------------+-------------------------------------
+-------------+
```

5. Vous pouvez interroger des tables dans l'unité de partage des données en spécifiant la base de données et le schéma du partage de données. Pour de plus amples informations, veuillez consulter Exemples de requêtes entre bases de données. Les requêtes suivantes renvoient les données de vente et de vendeurs à partir de la table SALES de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

```
SELECT * FROM myconsumer_db.myproducer_schema.tickit_sales_view;
```

```
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |       728 |
     109.2 | 2008-02-18 02:36:48 |
|       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |        76 |
      11.4 | 2008-06-06 05:00:16 |
|       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |       350 |
      52.5 | 2008-06-06 08:26:17 |
|       4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |       175 |
     26.25 | 2008-06-09 08:38:52 |
|       5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |       154 |
      23.1 | 2008-08-31 09:17:02 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

> **Note**
>
> La requête s'exécute sur la vue dans le schéma partagé. Vous ne pouvez pas vous connecter directement aux bases de données créées à partir d'unités de partage des données. Elles sont en lecture seule.

6. Pour exécuter une requête incluant des agrégations, utilisez l'exemple suivant.

```
SELECT * FROM myconsumer_db.myproducer_schema.tickit_sales ORDER BY 1,2 LIMIT 5;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|       1 |      1 |    36861 |   21191 |    7872 |   1875 |       4 |       728 |
     109.2 | 2008-02-18 02:36:48 |
|       2 |      4 |     8117 |   11498 |    4337 |   1983 |       2 |        76 |
      11.4 | 2008-06-06 05:00:16 |
|       3 |      5 |     1616 |   17433 |    8647 |   1983 |       2 |       350 |
      52.5 | 2008-06-06 08:26:17 |
```

```
|       4 |      5 |     1616 |   19715 |    8647 |   1986 |       1 |       175 |
     26.25 | 2008-06-09 08:38:52 |
|       5 |      6 |    47402 |   14115 |    8240 |   2069 |       2 |       154 |
      23.1 | 2008-08-31 09:17:02 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

La requête renvoie les données de vente et de vendeurs à partir des exemples de données TICKIT.

Pour obtenir des exemples de requêtes d'unités de partage des données supplémentaires, consultez Partage de l'accès en lecture aux données au sein d'un Compte AWS.

Types de partages de données dans Amazon Redshift

Une datashare est l'unité de partage de données dans Amazon Redshift. Utilisez les partages de données pour partager des données identiques Compte AWS ou différentes. Comptes AWS Vous pouvez également partager des données à des fins de lecture entre différents clusters Amazon Redshift.

Chaque unité de partage des données est associée à une base de données spécifique dans votre cluster Amazon Redshift.

Un administrateur producteur peut créer des partages de données et ajouter des objets de partage de données pour partager des données avec d'autres clusters, appelés partages sortants. Un administrateur client peut recevoir des partages de données provenant d'autres clusters, appelés partages entrants. Pour obtenir des détails sur les producteurs et les consommateurs, consultez Producteurs et consommateurs d'unités de partage des données.

Les objets Datashare sont des objets provenant de bases de données spécifiques d'un cluster que les administrateurs producteurs peuvent ajouter aux partages de données pour les partager avec les consommateurs de données. Les objets d'unité de partage des données sont en lecture seule pour les consommateurs de données. Les tables, vues et fonctions définies par l'utilisateur sont des exemples d'objets d'unité de partage des données. Vous pouvez ajouter des objets d'unité de partage des données à des unités de partage des données lors de la création ou de la modification d'une unité de partage des données à tout moment.

Le partage de données continue de fonctionner quand les clusters sont redimensionnés ou quand le cluster producteur est mis en pause.

Il existe différents types de partages de données : les partages de données standard, les partages de données et les partages AWS Data Exchange de données gérés. AWS Lake Formation Les pages suivantes fournissent un aperçu de chacun d'entre eux.

Unités de partage des données standard

Avec les partages de données standard, vous pouvez partager des données entre des clusters provisionnés, des groupes de travail sans serveur, des zones de disponibilité et. Comptes AWS Régions AWS Vous pouvez effectuer des partages entre différents types de cluster, mais aussi entre des clusters mis en service et Amazon Redshift sans serveur.

Pour partager les données, notez les identifiants des clusters mis en service, des espaces de noms sans serveur et des Compte AWS suivants :

- Les espaces de noms provisionnés sont des identifiants qui identifient les clusters provisionnés par Amazon Redshift. Lors de la création du cluster mis en service, un identifiant global unique (GUID) d'espace de noms est automatiquement créé et attaché au cluster. Un Amazon Resource Name (ARN) d'espace de noms se présente dans le format arn:{partition}:redshift:{region}:{account-id}:namespace:{namespace-guid}. Vous pouvez voir l'espace de noms d'un cluster Amazon Redshift sur la page de détails du cluster, dans la console Amazon Redshift.

  Dans le flux de travail de partage de données, la valeur GUID de l'espace de noms et l'ARN de l'espace de noms sont utilisés pour partager des données avec les clusters du. Compte AWS Vous pouvez également trouver l'espace de noms du cluster actuel à l'aide de la fonction `current_namespace`.
- Les Serverless namespaces (Espaces de noms sans serveur) sont des identifiants pour Amazon Redshift sans serveur. Un identifiant unique au niveau mondial (GUID) d'espace de noms est automatiquement créé lors de la création d'Amazon Redshift sans serveur et attaché à l'instance. Un ARN d'espace de noms sans serveur se présente dans le format arn:{partition}:redshift-serverless:{region}:{account-id}:namespace/{namespace-guid}.
- Comptes AWS peuvent être des consommateurs de partages de données et sont chacun représentés par un identifiant à 12 Compte AWS chiffres.

Pour les unités de partage des données standard, considérez ce qui suit :

- Lorsqu'un cluster producteur est supprimé, Amazon Redshift supprime les unités de partage des données créées par le cluster producteur. Lorsqu'un cluster producteur est sauvegardé et restauré, les unités de partage des données créées persistent sur le cluster restauré. Toutefois, les

autorisations d’unité de partage des données accordées à d’autres clusters ne sont plus valides sur le cluster restauré. Attribuez à nouveau les autorisations d’utilisation des unités de partage des données aux clusters consommateur souhaités. La base de données de consommateurs sur le cluster consommateur est tournée vers le datashare à partir du cluster d'origine où l'instantané est pris. Pour interroger les données partagées depuis le cluster restauré, l'administrateur du consommateur crée une base de données différente. L’administrateur peut également supprimer et recréer une base de données consommateur existante pour utiliser l’unité de partage des données à partir du cluster récemment restauré.

- Lorsqu’un cluster consommateur est supprimé et restauré à partir d’un instantané, l’accès précédent partagé avec ce cluster n’est plus valide ni visible. Si l'accès aux partages de données est toujours requis sur le cluster de consommateurs restauré, l'administrateur du producteur doit à nouveau autoriser l'utilisation des partages de données au cluster de consommateurs restauré. L'administrateur des consommateurs doit supprimer toutes les bases de données clients périmées créées à partir des partages de données inactifs. L’administrateur doit ensuite recréer la base de données consommateur à partir de l’unité de partage des données, une fois que le producteur a à nouveau accordé les autorisations. Comme le GUID de l'espace de noms est différent du cluster d'origine sur un cluster restauré, accordez à nouveau les autorisations de partage de données lorsque le cluster consommateur ou producteur est restauré à partir d'une sauvegarde.

AWS Data Exchange partages de données

Vous pouvez utiliser les AWS Data Exchange partages de données pour gérer la facturation du partage de données Amazon Redshift.

Un partage de AWS Data Exchange données est une unité de licence par laquelle vous pouvez partager vos données. AWS Data Exchange AWS gère toutes les facturations et tous les paiements associés aux abonnements AWS Data Exchange et à l'utilisation du partage de données Amazon Redshift. Les fournisseurs de données agréés peuvent ajouter des AWS Data Exchange partages de données aux AWS Data Exchange produits. Lorsque les clients s’abonnent à un produit avec des unités de partage des données AWS Data Exchange , ils ont accès aux unités de partage des données du produit.

AWS Data Exchange pour Amazon Redshift facilite l'accès sous licence à vos données Amazon Redshift via. AWS Data Exchange Lorsqu'un client s'abonne à un produit avec des partages de AWS Data Exchange données, il l'ajoute AWS Data Exchange automatiquement en tant que consommateur de données sur tous les partages de AWS Data Exchange données inclus dans le

produit. Les factures sont générées automatiquement et les paiements sont collectés de manière centralisée et décaissés automatiquement via AWS Marketplace Entitlement Service.

Les fournisseurs peuvent accorder des licences aux données dans Amazon Redshift à un niveau détaillé, tels que des schémas, des tables, des vues et des fonctions définies par l'utilisateur. Vous pouvez utiliser le même partage de AWS Data Exchange données sur plusieurs AWS Data Exchange produits. Tous les objets ajoutés au partage de AWS Data Exchange données sont accessibles aux consommateurs. Les producteurs peuvent consulter tous les AWS Data Exchange partages de données gérés en leur AWS Data Exchange nom à l'aide des opérations de l'API Amazon Redshift, des commandes SQL et de la console Amazon Redshift. Les clients qui s'abonnent aux partages de AWS Data Exchange données d'un produit ont un accès en lecture seule aux objets contenus dans les partages de données.

Les clients qui souhaitent consommer des données de producteurs tiers peuvent parcourir le catalogue AWS Data Exchange pour découvrir et s'abonner aux jeux de données dans Amazon Redshift. Une fois leur l'abonnement AWS Data Exchange actif, ils peuvent créer une base de données à partir de l'unité de partage des données dans leur cluster et interroger les données dans Amazon Redshift.

Comment fonctionnent les AWS Data Exchange partages de données

Gérer les AWS Data Exchange partages de données en tant qu'administrateur producteur

Si vous êtes un producteur de données (également appelé fournisseur AWS Data Exchange), vous pouvez créer des partages de AWS Data Exchange données qui se connectent à vos bases de données Amazon Redshift. Pour ajouter des AWS Data Exchange partages de données aux produits sur AWS Data Exchange, vous devez être un fournisseur enregistré AWS Data Exchange .

Pour plus d'informations sur la façon de démarrer avec les partages AWS Data Exchange de données, consultez. Partage de données Amazon Redshift sous licence sur AWS Data Exchange

Utilisation des AWS Data Exchange partages de données en tant que consommateur disposant d'un abonnement actif AWS Data Exchange

Si vous êtes un consommateur disposant d'un AWS Data Exchange abonnement actif (également appelé abonné activé AWS Data Exchange), vous pouvez parcourir le AWS Data Exchange catalogue sur la AWS Data Exchange console pour découvrir des produits contenant des partages de AWS Data Exchange données.

Après vous être abonné à un produit contenant des AWS Data Exchange partages de données, créez une base de données à partir du partage de données au sein de votre cluster. Vous pouvez

ensuite interroger les données dans Amazon Redshift directement sans extraire, transformer et charger les données.

Pour plus d'informations sur la façon de démarrer avec les partages AWS Data Exchange de données, consultez. Partage de données Amazon Redshift sous licence sur AWS Data Exchange

Pour les unités de partage des données AWS Data Exchange , considérez ce qui suit :

- Lorsqu'un cluster producteur est supprimé, Amazon Redshift supprime les unités de partage des données créées par le cluster producteur. Lorsqu'un cluster producteur est sauvegardé et restauré, les unités de partage des données créées persistent sur le cluster restauré. Pour que les abonnés aux données puissent continuer à accéder aux données, créez à nouveau les AWS Data Exchange partages de données et publiez-les dans les ensembles de données du produit. La base de données de consommateurs sur le cluster consommateur est tournée vers l'unité de partage des données à partir du cluster d'origine où l'instantané est pris. Pour interroger les données partagées à partir du cluster restauré, l'administrateur du consommateur crée une base de données différente ou supprime et recrée une base de données client existante afin d'utiliser le partage de AWS Data Exchange données nouvellement créé à partir du cluster récemment restauré.
- Lorsqu'un cluster de consommateurs est supprimé et restauré à partir d'un instantané, l'accès précédemment partagé à ce cluster reste valide et visible. L'administrateur du consommateur doit supprimer toutes les bases de données clients obsolètes créées à partir des partages de données inactifs et recréer la base de données clients à partir du partage de données une fois que le producteur a accordé à nouveau les autorisations. Comme le GUID de l'espace de noms est différent du cluster d'origine sur un cluster restauré, accordez à nouveau les autorisations de partage de données lorsque le cluster producteur est restauré à partir d'une sauvegarde.
- Nous vous recommandons de ne pas supprimer votre cluster si vous disposez d'unités de partage des données AWS Data Exchange . L'exécution de ce type de modification peut enfreindre les termes des produits de données dans AWS Data Exchange.

Producteurs et consommateurs d'unités de partage des données

Les producteurs de données (également appelés producteurs de partage de données ou producteurs de partage de données) sont des clusters à partir desquels vous souhaitez partager des données. Les administrateurs de producteurs et les propriétaires de bases de données peuvent créer des partages de données à l'aide de la commande CREATE DATASHARE. Vous pouvez ajouter des objets tels que des schémas, des tables, des vues et des fonctions SQL définies par l'utilisateur

(UDFs) à partir d'une base de données que vous souhaitez que le cluster de producteurs partage avec les clusters de consommateurs.

Les producteurs de données (également connus sous le nom de fournisseurs AWS Data Exchange) pour les partages AWS Data Exchange de données peuvent octroyer des licences de données via. AWS Data Exchange Les fournisseurs agréés peuvent ajouter des AWS Data Exchange partages de données aux AWS Data Exchange produits.

Lorsqu'un client s'abonne à un produit avec des partages de AWS Data Exchange données, il l'ajoute AWS Data Exchange automatiquement en tant que consommateur de données sur tous les partages de AWS Data Exchange données inclus dans le produit. AWS Data Exchange supprime également tous les clients des partages de AWS Data Exchange données à la fin de leur abonnement. AWS Data Exchange gère également automatiquement la facturation, le recouvrement des paiements et la distribution des paiements pour les produits payants grâce à des partages de AWS Data Exchange données. Pour de plus amples informations, veuillez consulter AWS Data Exchange partages de données. Pour vous inscrire en tant que fournisseur de AWS Data Exchange données, consultez Commencer en tant que fournisseur.

Les consommateurs de données (également appelés consommateurs de partage de données ou consommateurs d'unités de partage des données) sont des clusters qui reçoivent des unités de partage des données provenant de clusters producteur.

Les clusters Amazon Redshift qui partagent des données peuvent être identiques Régions AWS, différents Comptes AWS ou différents. Vous pouvez donc partager des données entre organisations et collaborer avec d'autres parties. Les administrateurs des consommateurs reçoivent les partages de données pour lesquels ils sont autorisés à utiliser et examinent le contenu de chaque partage de données. Pour consommer des données partagées, l'administrateur du consommateur crée une base de données Amazon Redshift à partir du partage de données. Il attribue ensuite des autorisations aux utilisateurs et aux rôle du cluster consommateur pour la base de données. Une fois les autorisations accordées, les utilisateurs et les rôles peuvent répertorier les objets partagés dans le cadre des requêtes de métadonnées standard, ainsi que les données locales du cluster consommateur. Ils peuvent commencer à interroger immédiatement.

Si vous êtes un consommateur disposant d'un AWS Data Exchange abonnement actif (également appelé « abonnés actifs » AWS Data Exchange), vous pouvez rechercher des up-to-date données, vous y abonner et les interroger de manière granulaire dans Amazon Redshift sans avoir à les extraire, à les transformer et à les charger. Pour de plus amples informations, veuillez consulter AWS Data Exchange partages de données.

Unités de partage des données gérées par AWS Lake Formation

Avec Amazon Redshift, vous pouvez accéder aux données en temps réel et les partager entre différents AWS comptes et clusters AWS Lake Formation Amazon Redshift par le biais de partages de données gérés. AWS Lake Formation les partages de données permettent aux fournisseurs de données de partager en toute sécurité les données en temps réel de leur lac de données Amazon S3 avec n'importe quel consommateur, y compris les autres AWS comptes et les clusters Amazon Redshift.

Vous pouvez ainsi définir et appliquer de manière centralisée les autorisations d'accès aux bases de données, aux tables, aux colonnes et aux lignes des partages de données Amazon Redshift et restreindre l'accès des utilisateurs aux objets d'un partage de données. AWS Lake Formation En partageant des données via Lake Formation, vous pouvez définir des autorisations dans Lake Formation et appliquer ces autorisations à n’importe quelle unité de partage des données et à ses objets. Par exemple, si vous disposez d’une table contenant des informations sur les employés, vous pouvez utiliser les filtres au niveau des colonnes de Lake Formation pour empêcher les employés qui ne travaillent pas dans le service des ressources humaines de voir des données d’identification personnelle (PII), telles qu’un numéro de sécurité sociale. Pour plus d’informations sur les filtres de données, consultez Filtrage des données et sécurité au niveau des cellules dans Lake Formation dans le Guide du développeur AWS Lake Formation .

Vous pouvez également utiliser les identifications dans Lake Formation pour configurer les autorisations sur les ressources de Lake Formation. Pour plus d’informations, consultez Contrôle d’accès basé sur l’identifiant de Lake Formation.

Amazon Redshift prend actuellement en charge le partage des données via Lake Formation au sein d’un même compte ou entre comptes. Le partage inter-régions n’est pas pris en charge actuellement.

Voici un aperçu général de l’utilisation de Lake Formation pour contrôler les autorisations des unités de partage des données :

1. Dans Amazon Redshift, l’administrateur du groupe de travail ou du cluster producteur crée une unité de partage des données sur ce dernier et autorise l’utilisation d’un compte Lake Formation.
2. L’administrateur du groupe de travail ou du cluster producteur autorise le compte Lake Formation à accéder à l’unité de partage des données.
3. L’administrateur Lake Formation découvre et enregistre les unités de partage des données. Ils doivent également découvrir les données AWS Glue ARNs auxquelles ils ont accès et associer les partages de données à un ARN AWS Glue Data Catalog . Si vous utilisez le, AWS CLI vous

pouvez découvrir et accepter les partages de données à l'aide des opérations de la CLI Redshift et. `describe-data-shares` `associate-data-share-consumer` Pour enregistrer une unité de partage des données, utilisez l'opération CLI `register-resource` de Lake Formation.

4. L'administrateur de Lake Formation crée une base de données fédérée dans le AWS Glue Data Catalog et configure les autorisations de Lake Formation afin de contrôler l'accès des utilisateurs aux objets du partage de données. Pour plus d'informations sur les bases de données fédérées dans AWS Glue, consultez la section Gestion des autorisations pour les données dans un partage de données Amazon Redshift.
5. L'administrateur de Lake Formation découvre les AWS Glue bases de données auxquelles il a accès et associe le partage de données à un ARN AWS Glue Data Catalog .
6. L'administrateur Redshift découvre la AWS Glue base de données à laquelle ARNs il a accès, crée une base de données externe dans le cluster de consommateurs Amazon Redshift à l'aide d'un ARN de base de données et en autorise l'utilisation aux utilisateurs de la AWS Glue base de données authentifiés avec des informations d'identification IAM pour commencer à interroger la base de données Amazon Redshift.
7. Les utilisateurs de base de données peuvent utiliser les vues SVV_EXTERNAL_TABLES et SVV_EXTERNAL_COLUMNS pour rechercher toutes les tables ou colonnes de la base de AWS Glue données auxquelles ils ont accès, puis ils peuvent interroger les tables de la base de données. AWS Glue
8. Lorsque l'administrateur du cluster de producteurs ou du groupe de travail décide de ne plus partager les données avec le cluster de consommateurs, l'administrateur de producteurs peut révoquer l'utilisation, annuler l'autorisation ou supprimer le partage de données de Redshift. Les autorisations et les objets associés dans Lake Formation ne sont pas automatiquement supprimés.

Pour plus d'informations sur le partage d'un partage de données en AWS Lake Formation tant qu'administrateur d'un cluster de producteurs ou d'un groupe de travail, consultez. Utilisation d'unités de partage des données gérées par Lake Formation en tant que producteur Pour utiliser les données partagées depuis un cluster producteur ou un groupe de travail, consultez Utilisation d'unités de partage des données gérées par Lake Formation en tant que consommateur.

Valeurs de statut du partage de données dans Amazon Redshift

Avec Amazon Redshift, vous pouvez partager des données en temps réel en toute sécurité entre des clusters Amazon Redshift sans avoir à copier ou à transférer des données. Datashares for Amazon

Redshift vous permet de partager les résultats de requêtes en direct, y compris les mises à jour des données sources, avec n'importe quel cluster Amazon Redshift appartenant à des comptes et régions identiques ou différents. AWS AWS Cette rubrique décrit les statuts possibles que les partages de données peuvent avoir dans Amazon Redshift.

Avec les unités de partage des données entre comptes, il existe différents statuts des unités de partage des données qui nécessitent votre intervention. Votre partage de données peut avoir le statut « actif », « action requise » ou « inactif ».

Le paragraphe ci-dessous décrit chaque statut d'unité de partage des données et les actions requises :

- Lorsqu'un administrateur de producteurs crée un partage de données, le statut du partage de données sur le cluster de producteurs est En attente d'autorisation. L'administrateur du producteur peut autoriser les consommateurs de données à accéder au partage de données. Aucune action n'est prévue pour l'administrateur du consommateur.
- Lorsqu'un administrateur du producteur autorise le partage de données, le statut du partage de données devient Autorisé sur le cluster de producteurs. Aucune action n'est prévue pour l'administrateur du producteur. Lorsqu'il existe au moins une association avec un consommateur de données pour l'unité de partage des données, l'état de l'unité de partage des données passe de Authorized (Autorisé) à Active (Actif).

  L'état de partage de l'unité de partage des données passe alors à Available (Action required on the Amazon Redshift console) (Disponible (Action requise sur la console Amazon Redshift)) sur le cluster consommateur. L'administrateur du consommateur peut associer le partage de données aux consommateurs de données ou rejeter le partage de données. L'administrateur du consommateur peut également utiliser la AWS CLI commande `describeDatashareforConsumer` pour afficher l'état des partages de données. L'administrateur peut également utiliser la commande CLI `describeDatashare` et fournir le nom Amazon Resource Name (ARN) de l'unité de partage des données pour afficher l'état de l'unité de partage des données.
- Lorsque l'administrateur du consommateur associe un partage de données à des consommateurs de données, le statut du partage de données devient actif sur le cluster de producteurs. Lorsqu'il existe au moins une association avec un consommateur de données pour l'unité de partage des données, l'état de l'unité de partage des données passe de Authorized (Autorisé) à Active (Actif). Aucune action n'est requise de la part de l'administrateur du producteur.

L'état de l'unité de partage des données devient Active (Actif) sur le cluster consommateur. Aucune action n'est requise de la part de l'administrateur du consommateur.

- Lorsque l'administrateur du consommateur supprime une association de consommateurs d'un partage de données, le statut du partage de données devient actif ou autorisé. Il devient Active (Actif) lorsqu'il existe au moins une association pour l'unité de partage des données avec un autre consommateur de données. Il devient Authorized (Autorisé) lorsqu'il n'existe pas d'association de consommateurs avec l'unité de partage des données sur le cluster producteur. Aucune action n'est prévue pour l'administrateur du producteur.

  L'état de l'unité de partage des données passe à Action required (Action requise) sur le cluster consommateur si toutes les associations sont supprimées. L'administrateur du consommateur peut réassocier un partage de données aux consommateurs de données lorsque le partage de données est disponible pour les consommateurs.
- Lorsqu'un administrateur client refuse un partage de données, le statut du partage de données sur le cluster de producteurs devient Action requise et Refusé sur le cluster de consommateurs. L'administrateur du producteur peut réautoriser le partage de données. Aucune action n'est prévue pour l'administrateur du consommateur.
- Lorsque l'administrateur du producteur supprime l'autorisation d'un partage de données, le statut du partage de données devient Action requise sur le cluster de producteurs. L'administrateur du producteur peut choisir de réautoriser le partage de données, si nécessaire. Aucune action n'est requise de la part de l'administrateur du consommateur.

Gérer l'accès aux opérations d'API d'unités de partage de données avec les politiques IAM

Pour contrôler l'accès aux opérations d'API de partage de données, utilisez les politiques IAM basées sur les actions. Pour plus d'informations sur la gestion des politiques IAM, consultez Gestion des politiques IAM dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les autorisations requises pour utiliser les opérations d'API de partage de données, consultez Permissions required to use the data sharing API operations dans le Guide de gestion Amazon Redshift.

Pour sécuriser davantage le partage de données entre comptes, vous pouvez utiliser une clé conditionnelle `ConsumerIdentifier` pour les opérations d'API `AuthorizeDataShare` et

`DeauthorizeDataShare`. Ce faisant, vous pouvez contrôler de manière explicite qui Comptes AWS peut appeler les deux opérations d'API.

Vous pouvez refuser l'autorisation ou l'annulation du partage de données pour tout consommateur qui n'est pas votre propre compte. Pour ce faire, spécifiez le Compte AWS numéro dans la politique IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Deny",
            "Action": [
                "redshift:AuthorizeDataShare",
                "redshift:DeauthorizeDataShare"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "redshift:ConsumerIdentifier": "555555555555"
                }
            }
        }
    ]
}
```

Vous pouvez autoriser un producteur possédant un DataShareArn **`testshare2`** à partager explicitement avec un consommateur dont le numéro est 111122223333 dans Compte AWS la politique IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "redshift:AuthorizeDataShare",
                "redshift:DeauthorizeDataShare"
            ],
```

```
            "Resource": "arn:aws:redshift:us-
east-1:666666666666:datashare:af06285e-8a45-4ee9-b598-648c218c8ff1/testshare2",
            "Condition": {
                "StringEquals": {
                    "redshift:ConsumerIdentifier": "111122223333"
                }
            }
        }
    ]
}
```

Connexion aux bases de données des consommateurs dans Amazon Redshift

Avec une connexion directe à une base de données de partage de données, vous pouvez vous connecter directement à une base de données créée à partir d'un partage de données de la même manière que vous pouvez vous connecter à tout autre type de base de données Amazon Redshift. Par exemple, vous pouvez vous connecter à une base de données créée à partir d'un partage de données à l'aide de pilotes JDBC ou ODBC, de l'éditeur de requêtes Amazon Redshift v2 ou de tout autre outil permettant de se connecter à une base de données Amazon Redshift. Pour plus d'informations, consultez Connexion à un entrepôt de données Amazon Redshift à l'aide des outils client SQL.

Accéder aux données partagées

Lorsque vous vous connectez à une base de données créée à partir d'un partage de données, vous pouvez interroger les objets partagés à l'aide d'une notation en deux parties (). *schema_name*.*table_name* Si la table se trouve dans le chemin de recherche de la base de données des consommateurs, vous pouvez également utiliser la notation en une partie (*table_name*).

Si vous souhaitez effectuer des requêtes entre bases de données, vous pouvez utiliser la notation en trois parties ()*consumer_database_name*.*schema_name*.*table_name*. Ces requêtes peuvent faire référence à des objets partagés provenant d'une autre base de données de consommateurs du cluster ou à des objets locaux provenant de bases de données locales. Ils peuvent également faire référence à la fois à des bases de données locales et à des données partagées depuis d'autres clusters au sein de la même requête.

> **Note**
>
> Une base de données créée à partir d'un partage de données ne possède pas de catalogue local. Par conséquent, toutes les requêtes qui accèdent aux tables de catalogue locales, par exemple`pg_class`, renvoient un résultat vide.

Accès aux métadonnées pour les objets partagés

Pour aider les administrateurs de clusters à découvrir les objets partagés dans la base de données des consommateurs, Amazon Redshift fournit un ensemble de vues de métadonnées et de commandes SHOW répertoriant les métadonnées de ces objets. Lorsque vous vous connectez à une base de données destinée aux consommateurs, ces vues et commandes de métadonnées ne prennent pas en charge la découverte de métadonnées entre bases de données. Ils renvoient uniquement les métadonnées des objets partagés du partage de données associés à la base de données connectée.

Utilisez SHOW SCHEMAS pour afficher la liste des schémas partagés dans le partage de données associé à la base de données connectée. Pour de plus amples informations, veuillez consulter SHOW SCHEMAS.

Utilisez SHOW TABLES pour afficher la liste des tables d'un schéma partagé à partir du partage de données associé à la base de données connectée. Pour de plus amples informations, veuillez consulter SHOW TABLES.

Utilisez SHOW COLUMNS pour afficher la liste des colonnes d'une table partagée dans le partage de données associé à la base de données connectée. Pour de plus amples informations, veuillez consulter SHOW COLUMNS.

Utilisez SVV_ALL_SCHEMAS pour afficher la liste des schémas partagés dans le partage de données associé à la base de données connectée. Pour de plus amples informations, veuillez consulter SVV_ALL_SCHEMAS.

Utilisez SVV_ALL_TABLES pour afficher la liste des tables partagées dans le partage de données associé à la base de données connectée. Pour de plus amples informations, veuillez consulter SVV_ALL_TABLES.

Utilisez SVV_ALL_COLUMNS pour afficher la liste des colonnes partagées dans le partage de données associé à la base de données connectée. Pour de plus amples informations, veuillez consulter SVV_ALL_COLUMNS.

Intégrer le partage de données Amazon Redshift aux outils de business intelligence

Pour intégrer le partage de données aux outils de Business Intelligence (BI), nous vous recommandons d'utiliser les pilotes Amazon Redshift JDBC ou ODBC. Les pilotes JDBC et ODBC Amazon Redshift prennent en charge le fonctionnement de l'`GetCatalogs`API dans les pilotes. Cette opération renvoie une liste de toutes les bases de données, y compris celles créées à partir de partages de données.

Les pilotes prennent également en charge les opérations en aval, telles que `GetSchemas` et GetTables, qui renvoient des données provenant de toutes les bases de données `GetCatalogs` renvoyées. Les chauffeurs fournissent cette assistance même lorsque vous ne spécifiez pas explicitement le catalogue dans l'appel. Pour plus d'informations sur les pilotes JDBC ou ODBC, consultez la section Configuration des connexions dans le guide de gestion Amazon Redshift.

L'éditeur de requêtes Amazon Redshift v2 inclut des bases de données clients dans son interface de commutation de connexion. Cependant, la plupart des outils excluent ces bases de données et n'incluent que les bases de données de clusters locales en tant que bases de données connectables.

> **Note**
>
> Une nouvelle base de données système nommée `sys:internal` a été ajoutée pour la maintenance interne. Certains outils incluent cette base de données système en tant que base de données connectable. Cependant, vous ne pouvez pas vous y connecter ni exécuter de requêtes sur ses objets.

Surveillance et audit du partage des données dans Amazon Redshift

Avec Amazon Redshift, vous pouvez surveiller et auditer les activités de partage de données afin de garantir la conformité et la sécurité.

En effectuant l'audit du partage de données, les producteurs peuvent suivre l'évolution de l'unité de partage des données. Par exemple, l'audit permet de suivre le moment où des partages de données sont créés, des objets sont ajoutés ou supprimés, et des autorisations sont accordées ou révoquées aux clusters, AWS comptes ou régions Amazon Redshift. AWS

En plus de l’audit, les producteurs et les consommateurs suivent l’utilisation des unités de partage des données à plusieurs niveaux de précision : au niveau du compte, du cluster et de l’objet. Pour plus d’informations sur le suivi de l’utilisation et l’audit des vues, consultez SVL_DATASHARE_CHANGE_LOG et SVL_DATASHARE_USAGE_PRODUCER.

Vous pouvez surveiller les unités de partage des données en interrogeant les vues système.

1. L'administrateur du producteur qui souhaite partager des données crée un partage de données Amazon Redshift. L'administrateur du producteur ajoute ensuite les objets de base de données nécessaires. Il peut s’agir de schémas, de tables et de vues de l’unité de partage des données et spécifie une liste de consommateurs avec lesquels les objets seront partagés.

   Utilisez les vues système suivantes pour obtenir des vues consolidées permettant de suivre les modifications et l'utilisation des partages de données sur les clusters de producteurs et de and/or consommateurs :

   - SYS_DATASHARE_CHANGE_LOG
   - SYS_DATASHARE_USAGE_CONSUMER
   - SYS_DATASHARE_USAGE_PRODUCER

   Utilisez les vues système suivantes pour afficher les objets d’unité de partage des données et les informations sur les consommateurs de données pour les unités de partage des données sortantes :

   - SVV_DATASHARES
   - SVV_DATASHARE_CONSUMERS
   - SVV_DATASHARE_OBJECTS
2. Les administrateurs des consommateurs examinent les partages de données pour lesquels ils sont autorisés à utiliser et examinent le contenu de chaque partage de données en consultant les partages de données entrants à l'aide de. SVV_DATASHARES

   Pour utiliser les données partagées, chaque administrateur client crée une base de données Amazon Redshift à partir du partage de données. L’administrateur attribue ensuite des autorisations aux utilisateurs et rôles appropriés dans le cluster consommateur. Les utilisateurs et les groupes peuvent répertorier les objets partagés dans le cadre des requêtes de métadonnées standard en affichant les vues système de métadonnées suivantes, et peuvent commencer immédiatement à interroger les données.

   - SVV_REDSHIFT_COLUMNS

- SVV_REDSHIFT_DATABASES
- SVV_REDSHIFT_FUNCTIONS
- SVV_REDSHIFT_SCHEMAS
- SVV_REDSHIFT_TABLES

Pour afficher les objets des schémas locaux et partagés et des schémas externes Amazon Redshift, utilisez les vues système de métadonnées suivantes pour les interroger.

- SVV_ALL_COLUMNS
- SVV_ALL_SCHEMAS
- SVV_ALL_TABLES

Lorsque vous vous connectez à une base de données de consommateurs, la découverte entre bases de données est désactivée. Les vues du système de métadonnées renvoient uniquement des métadonnées pour les objets partagés dans le partage de données associé à la base de données connectée.

Intégrer le partage de données Amazon Redshift avec AWS CloudTrail

Le partage de données est intégré à AWS CloudTrail. CloudTrail est un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans Amazon Redshift. CloudTrail capture tous les appels d'API pour le partage de données sous forme d'événements. Les appels capturés incluent les appels de la console AWS CloudTrail et les appels de code aux opérations de partage de données. Pour plus d'informations sur l'intégration d'Amazon Redshift avec AWS CloudTrail, consultez Logging with. CloudTrail

Pour plus d'informations CloudTrail, consultez la section CloudTrail Fonctionnement.

Données semi-structurées dans Amazon Redshift

En utilisant la prise en charge des données semi-structurées dans Amazon Redshift, vous pouvez ingérer et stocker des données semi-structurées dans vos entrepôts des données Amazon Redshift. En utilisant le type de données SUPER et le langage PartiQL, Amazon Redshift étend les capacités de l'entrepôt des données pour s'intégrer aux sources de données SQL et NoSQL. Ainsi, Amazon Redshift permet des analyses efficaces sur des données relationnelles et semi-structurées stockées telles que JSON.

Amazon Redshift offre deux formes de prise en charge des données semi-structurées : le type de données SUPER et Amazon Redshift Spectrum.

Utilisez le type de données SUPER si vous devez insérer ou mettre à jour de petits lots de données JSON avec une faible latence. Utilisez également SUPER lorsque votre requête nécessite une cohérence forte, des performances de requête prévisibles, une prise en charge des requêtes complexes et une facilité d'utilisation avec des schémas évolutifs et des données sans schéma.

En revanche, utilisez Amazon Redshift Spectrum avec un format de fichier ouvert si votre requête de données nécessite une intégration avec AWS d'autres services et avec des données principalement stockées dans Amazon S3 à des fins d'archivage.

Cas d'utilisation pour le type de données SUPER

La prise en charge des données semi-structurées à l'aide du type de données SUPER dans Amazon Redshift offre des performances, une flexibilité et une facilité d'utilisation supérieures. Les cas d'utilisation suivants aident à illustrer comment utiliser un support de données semi-structurées avec SUPER.

Insertion rapide et flexible de données JSON – Amazon Redshift prend en charge les transactions rapides qui peuvent analyser JSON et le stocker sous forme de valeur SUPER. Les transactions d'insertion peuvent opérer jusqu'à cinq fois plus vite que les mêmes insertions dans des tables qui ont fragmenté les attributs de SUPER en colonnes conventionnelles. Par exemple, supposons que le JSON entrant soit de la forme {« a » :…, « b » :…, « c » :..., etc.}. Vous pouvez considérablement accélérer les performances d'insertion en stockant le JSON entrant dans une table TJ avec une seule colonne SUPER S, au lieu de le stocker dans une table conventionnelle TR avec des colonnes « a », « b », « c », et ainsi de suite. Lorsqu'il y a des centaines d'attributs dans le JSON, l'avantage de performances du type de données SUPER devient substantiel.

En outre, le type de données SUPER n'a pas besoin d'un schéma régulier. Vous n'avez pas besoin d'analyser et de nettoyer le JSON entrant avant de le stocker. Par exemple, supposons qu'un fichier JSON entrant possède un attribut « c » de type chaîne de caractères et d'autres qui possèdent un attribut « c » de type entier, sans le type de données SUPER. Dans ce cas, vous devez soit séparer les colonnes c_string et c_int, soit nettoyer les données. En revanche, avec le type de données SUPER, toutes les données JSON sont stockées pendant l'ingestion sans perte d'informations. Plus tard, vous pouvez utiliser l'extension PartiQL de SQL pour analyser les informations.

Requêtes flexibles pour découverte – Après avoir stocké vos données semi-structurées (telles que JSON) dans une valeur de données SUPER, vous pouvez les interroger sans imposer de schéma. Vous pouvez utiliser le typage dynamique et la sémantique laxiste de PartiQL pour exécuter vos requêtes et découvrir les données profondément imbriquées dont vous avez besoin, sans avoir à imposer un schéma avant la requête.

Requêtes flexibles pour les opérations d'extraction, de chargement, de transformation (ETL) en vues matérialisées conventionnelles – Après avoir stocké vos données non schématiques et semi-structurées dans SUPER, vous pouvez utiliser les vues matérialisées PartiQL pour introspecter les données et les fragmenter en vues matérialisées.

Les vues matérialisées avec les données fragmentées sont un bon exemple d'avantages en termes de performances et de convivialité pour vos cas d'analyse classiques. Lorsque vous effectuez des analyses sur les données fragmentées, l'organisation en colonnes des vues matérialisées Amazon Redshift offre de meilleures performances. En outre, les utilisateurs et les outils de Business Intelligence (BI) qui exigent un schéma conventionnel pour les données ingérées peuvent utiliser des vues (matérialisées ou virtuelles) comme présentation conventionnelle des données.

Une fois que vos vues matérialisées PartiQL ont extrait les données trouvées dans JSON ou SUPER dans des vues matérialisées en colonnes conventionnelles, vous pouvez interroger les vues matérialisées. Pour plus d'informations sur le fonctionnement du type de données SUPER avec les vues matérialisées, consultez Type de données SUPER et vues matérialisées.

Vous pouvez appliquer des politiques de masquage dynamique des données aux valeurs `scalar` figurant sur les chemins des colonnes de type SUPER. Pour plus d'informations sur le masquage dynamique des données, consultez Masquage dynamique des données. Pour plus d'informations sur l'utilisation du masquage dynamique des données avec le type de données SUPER, consultez Utilisation du masquage dynamique des données avec des chemins de type de données SUPER (version préliminaire).

Pour plus d'informations sur le type de données SUPER, consultez Type SUPER.

Pour obtenir des exemples de l'utilisation du type de données SUPER, consultez les sous-sections de cette rubrique, en commençant par Jeu de données échantillon SUPER.

Concepts pour l'utilisation des types de données SUPER

Vous trouverez ci-après quelques concepts de types de données Amazon Redshift SUPER.

Comprendre le type de données SUPER dans Amazon Redshift – Le type de données SUPER est un type de données Amazon Redshift qui permet le stockage de tableaux et de structures sans schéma qui contiennent des scalaires Amazon Redshift et éventuellement des tableaux et structures imbriqués. Le type de données SUPER peut stocker nativement différents formats de données semi-structurées, tels que JSON ou des données provenant de sources orientées document. Vous pouvez ajouter une nouvelle colonne SUPER pour stocker des données semi-structurées et écrire des requêtes qui accèdent à la colonne SUPER, ainsi que les colonnes scalaires habituelles. Pour plus d'informations sur le type de données SUPER, consultez Type SUPER.

Ingérer JSON sans schéma dans SUPER – Grâce à la souplesse du type de données semi-structurées SUPER, Amazon Redshift peut recevoir et intégrer des JSON sans schéma dans une valeur SUPER. Par exemple, Amazon Redshift peut ingérer la valeur JSON [10.5, « first »] dans une valeur SUPER [10.5, « first »], c'est-à-dire un tableau contenant un decimal 10.5 et un varchar « first » Amazon Redshift. Amazon Redshift peut ingérer le JSON dans une valeur SUPER en utilisant l'instruction COPY ou la fonction JSON parse, telle que json_parse('[10.5, « first »]'). COPY et `json_parse` ingèrent tous deux JSON en utilisant une sémantique d'analyse stricte par défaut. Vous pouvez également construire des valeurs SUPER, y compris des tableaux et des structures, en utilisant les données de base de données elles-mêmes.

La colonne SUPER ne nécessite aucune modification de schéma lors de l'ingestion des structures irrégulières de JSON sans schéma. Par exemple, lors de l'analyse d'un flux de clics, vous stockez d'abord dans la colonne SUPER des structures « click » avec les attributs « IP » et « time ». Vous pouvez ajouter un attribut « ID client » sans modifier votre schéma afin d'ingérer de telles modifications.

Le format natif utilisé pour le type de données SUPER est un format binaire qui nécessite moins d'espace que la valeur JSON sous sa forme textuelle. Cela permet d'accélérer l'ingestion et le traitement en cours d'exécution des valeurs SUPER lors des requêtes.

Interrogez les données SUPER avec partiQL — partiQL est une extension rétrocompatible de SQL-92 que de nombreux services utilisent actuellement. AWS Grâce à PartiQL, les constructions

SQL familières combinent de manière transparente l'accès aux données SQL classiques, sous forme de tableaux, et aux données semi-structurées de SUPER. Vous pouvez effectuer la navigation dans les objets et les tableaux et désimbriquer les tableaux. PartiQL étend le langage SQL standard pour exprimer et traiter de manière déclarative les données imbriquées et multivaluées.

PartiQL est une extension de SQL où les données imbriquées et non schématiques des colonnes SUPER sont des citoyens de première classe. PartiQL n'exige pas que toutes les expressions de requêtes soient vérifiées par type lors de la compilation de la requête. Cette approche permet aux expressions de requêtes qui contiennent le type de données SUPER d'être typées dynamiquement pendant l'exécution de la requête lorsque les types réels des données contenues dans les colonnes SUPER sont accédés. De plus, PartiQL fonctionne dans un mode laxiste dans lequel les incohérences de type ne provoquent pas d'échecs, mais renvoient null. La combinaison du traitement des requêtes sans schéma et laxiste rend PartiQL idéal pour les applications d'extraction, de chargement et de transfert (ELT) pour lesquelles votre requête SQL évalue les données JSON qui sont ingérées dans les colonnes SUPER.

Intégration avec Redshift Spectrum – Amazon Redshift prend en charge plusieurs aspects de PartiQL lors de l'exécution de requêtes Redshift Spectrum sur JSON, Parquet et d'autres formats comportant des données imbriquées. Redshift Spectrum ne supporte que les données imbriquées qui ont des schémas. Par exemple, avec Redshift Spectrum, vous pouvez déclarer que vos données JSON ont un attribut nested_schemaful_example dans un schéma ARRAY<STRUCT<a:INTEGER, b:DECIMAL(5,2)>>. Le schéma de cet attribut détermine que les données contiennent toujours un tableau, qui contient une structure avec un nombre entier a et un nombre décimal b. Si les données changent pour inclure plus d'attributs, le type change également. En revanche, le type de données SUPER ne nécessite aucun schéma. Vous pouvez stocker des tableaux avec des éléments de structure ayant des attributs ou des types différents. En outre, certaines valeurs peuvent être stockées en dehors des tableaux.

Pour plus d'informations sur les fonctions prenant en charge le type de données SUPER, consultez les rubriques suivantes :

- Fonction ABS
- Fonction CEILING (ou CEIL)
- Fonction FLOOR
- Fonction ROUND
- Fonction SIGN
- Fonction TRUNC

Considérations relatives aux données Super

Lorsque vous travaillez avec des données SUPER, tenez compte des points suivants :

- Utilisez le pilote JDBC version 1.2.50, le pilote ODBC version 1.4.17 ou ultérieure, et le pilote Amazon Redshift Python version 2.0.872 ou ultérieure.

  Pour plus d'informations sur les pilotes JDBC, consultez Configuration d'une connexion JDBC.

  Pour plus d'informations sur les pilotes ODBC, consultez Configuration d'une connexion ODBC.
- Retrouvez les exemples de schémas utilisés dans les rubriques suivantes à l'adresse Jeu de données échantillon SUPER.
- Tous les exemples de code SQL utilisés dans les rubriques suivantes sont inclus avec le même préfixe S3 pour le téléchargement. Ceux-ci comprennent le langage de définition de données (DDL) et les instructions COPY, ainsi que certaines requêtes modifiées TPC-H qui fonctionnent avec SUPER.

  Pour afficher ou télécharger les fichiers SQL, procédez de l'une des manières suivantes :

  - Téléchargez le fichier SQL du tutoriel SUPER et le fichier TPC-H.
  - En utilisant la CLI Amazon S3, exécutez la commande suivante. Vous pouvez utiliser votre propre chemin d'accès.

```
aws s3 cp s3://redshift-downloads/semistructured/tutorialscripts/semistructured-
tutorial.sql /target/path
aws s3 cp s3://redshift-downloads/semistructured/tutorialscripts/
super_tpch_queries.sql /target/path
```

Pour plus d'informations sur les configurations SUPER, consultez Configurations SUPER.

Jeu de données échantillon SUPER

Vous pouvez explorer et analyser l'exemple de jeu de données SUPER, qui contient des données relatives aux ventes de produits fictifs dans différentes catégories, régions et périodes. Les sections suivantes fournissent des informations sur l'accès, l'interrogation et la manipulation du jeu de données SUPER sample au sein de votre cluster Amazon Redshift.

Le schéma de table et le modèle de données utilisés pour les exemples d'ingestion et de requête sont définis comme suit.

```
/*customer-orders-lineitem*/
CREATE TABLE customer_orders_lineitem
(c_custkey bigint
,c_name varchar
,c_address varchar
,c_nationkey smallint
,c_phone varchar
,c_acctbal decimal(12,2)
,c_mktsegment varchar
,c_comment varchar
,c_orders super
);

/* Datamodel of documents to be stored in c_orders Super column would be as follows*/
ARRAY < STRUCT < o_orderkey:bigint
                         ,o_orderstatus:string
                         ,o_totalprice:double
                         ,o_orderdate:string
                         ,o_orderpriority:string
                         ,o_clerk:string
                         ,o_shippriority:int
                         ,o_comment:string
                         ,o_lineitems:ARRAY < STRUCT < l_partkey:bigint
                                                      ,l_suppkey:bigint
                                                      ,l_linenumber:int
                                                      ,l_quantity:double
                                                      ,l_extendedprice:double
                                                      ,l_discount:double
                                                      ,l_tax:double
                                                      ,l_returnflag:string
                                                      ,l_linestatus:string
                                                      ,l_shipdate:string
                                                      ,l_commitdate:string
                                                      ,l_receiptdate:string
                                                      ,l_shipinstruct:string
                                                      ,l_shipmode:string
                                                      ,l_comment:string
                                                      > >
                          > >
```

```
/*part*/
CREATE TABLE part
(
 p_partkey bigint
 ,p_name varchar
 ,p_mfgr varchar
 ,p_brand varchar
 ,p_type varchar
 ,p_size int
 ,p_container varchar
 ,p_retailprice decimal(12,2)
 ,p_comment varchar
 );

/*region-nations*/
CREATE TABLE region_nations
(
 r_regionkey smallint
 ,r_name varchar
 ,r_comment varchar
 ,r_nations super
 );

 /* Datamodel of documents to be stored in r_nations Super column would be as follows*/
 ARRAY < STRUCT < n_nationkey:int,n_name:string,n_comment:string > >

/*supplier-partsupp*/
CREATE TABLE supplier_partsupp
(
 s_suppkey bigint
 ,s_name varchar
 ,s_address varchar
 ,s_nationkey smallint
 ,s_phone varchar
 ,s_acctbal double precision
 ,s_comment varchar
 ,s_partsupps super
 );

 /* Datamodel of documents to be stored in s_partsupps Super column would be as
 follows*/
 ARRAY < STRUCT <
 ps_partkey:bigint,ps_availqty:int,ps_supplycost:double,ps_comment:string > >
```

Chargement de données semi-structurées dans Amazon Redshift

Utilisez le type de données SUPER pour persister et interroger des données hiérarchiques et génériques dans Amazon Redshift. Amazon Redshift introduit la fonction `json_parse` pour analyser les données au format JSON et les convertir en représentation SUPER. Amazon Redshift prend également en charge le chargement des colonnes SUPER via l'instruction COPY. Les formats de fichiers pris en charge sont JSON, Avro, texte, format CSV (valeurs séparées par des virgules), Parquet et ORC.

Pour en savoir plus sur les tables utilisées dans les exemples suivants, consultez Jeu de données échantillon SUPER.

Pour plus d'informations sur la fonction `json_parse`, consultez Fonction JSON_PARSE.

L'encodage par défaut pour le type de données SUPER est ZSTD.

Analyse de documents JSON en colonnes SUPER

Vous pouvez insérer ou mettre à jour des données JSON dans une colonne SUPER à l'aide de la fonction `json_parse`. La fonction analyse les données au format JSON et les convertit en type de données SUPER que vous pouvez utiliser dans les instructions INSERT ou UPDATE.

L'exemple suivant insère des données JSON dans une colonne SUPER. Si la fonction `json_parse` est absente de la requête, Amazon Redshift traite la valeur comme une chaîne unique au lieu d'une chaîne formatée JSON qui doit être analysée.

Si vous mettez à jour une colonne de données SUPER, Amazon Redshift exige que le document complet soit transmis aux valeurs de colonne. Amazon Redshift ne prend pas en charge les mises à jour partielles.

```
INSERT INTO region_nations VALUES(0,
   'lar deposits. blithely final packages cajole. regular waters are final requests.
 regular accounts are according to',
   'AFRICA',
   JSON_PARSE('{"r_nations":[
      {"n_comment":" haggle. carefully final deposits detect slyly agai",
         "n_nationkey":0,
         "n_name":"ALGERIA"
      },
      {"n_comment":"ven packages wake quickly. regu",
         "n_nationkey":5,
```

```
         "n_name":"ETHIOPIA"
      },
      {"n_comment":" pending excuses haggle furiously deposits. pending, express pinto
 beans wake fluffily past t",
         "n_nationkey":14,
         "n_name":"KENYA"
      },
      {"n_comment":"rns. blithely bold courts among the closely regular packages use
 furiously bold platelets?",
         "n_nationkey":15,
         "n_name":"MOROCCO"
      },
      {"n_comment":"s. ironic, unusual asymptotes wake blithely r",
         "n_nationkey":16,
         "n_name":"MOZAMBIQUE"
      }
   ]
}'));
```

Utilisation de COPY pour charger des colonnes SUPER dans Amazon Redshift

Dans les sections suivantes, vous pouvez découvrir les différentes façons d'utiliser l'instruction COPY pour charger des données JSON dans Amazon Redshift.

Copie de données à partir de JSON et Avro

En utilisant la prise en charge des données semi-structurées dans Amazon Redshift, vous pouvez charger un document JSON sans fragmenter les attributs de ses structures JSON en plusieurs colonnes.

Amazon Redshift fournit deux méthodes pour ingérer un document JSON en utilisant COPY, même avec une structure JSON totalement ou partiellement inconnue :

1. Stocker les données dérivées d'un document JSON dans une seule colonne de données SUPER en choisissant l'option `noshred`. Cette méthode est utile lorsque le schéma n'est pas connu ou est censé changer. Ainsi, cette méthode facilite le stockage du tuple entier dans une seule colonne SUPER.
2. Déchiquetez le document JSON en plusieurs colonnes Amazon Redshift en choisissant l'option `auto` ou `jsonpaths`. Les attributs peuvent être des scalaires Amazon Redshift ou des valeurs SUPER.

Vous pouvez utiliser ces options avec les formats JSON ou Avro.

La taille maximale d'un objet JSON avant le déchiquetage est de 4 Mo.

Copie d'un document JSON dans une seule colonne de données SUPER

Pour copier un document JSON dans une seule colonne de données SUPER, créez une table avec une seule colonne de données SUPER.

```
CREATE TABLE region_nations_noshred (rdata SUPER);
```

Copiez les données d'Amazon S3 dans la colonne de données SUPER unique. Pour ingérer les données source JSON dans une seule colonne de données SUPER, spécifiez l'option `noshred` dans la clause FORMAT JSON.

```
COPY region_nations_noshred FROM 's3://redshift-downloads/semistructured/tpch-nested/
data/json/region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 'noshred';
```

Une fois que COPY a réussi à ingérer le JSON, votre table a une colonne de données SUPER `rdata` qui contient les données de l'objet JSON entier. Les données ingérées conservent toutes les propriétés de la hiérarchie JSON. Toutefois, les feuilles sont converties en types scalaires Amazon Redshift pour un traitement efficace des requêtes.

Utilisez la requête suivante pour récupérer la chaîne JSON originale.

```
SELECT rdata FROM region_nations_noshred;
```

Lorsque Amazon Redshift génère une colonne de données SUPER, elle devient accessible à l'aide de JDBC en tant que chaîne de caractères par sérialisation JSON. Pour de plus amples informations, veuillez consulter Sérialisation de JSON imbriqué complexe.

Copier un document JSON dans plusieurs colonnes de données SUPER

Vous pouvez fragmenter un document JSON en plusieurs colonnes, qui peuvent être des colonnes de données SUPER ou des types scalaires Amazon Redshift. Amazon Redshift répartit différentes portions de l'objet JSON dans différentes colonnes.

```
CREATE TABLE region_nations
(
```

```
 r_regionkey smallint
 ,r_name varchar
 ,r_comment varchar
 ,r_nations super
 );
```

Pour copier les données de l'exemple précédent dans la table, spécifiez l'option AUTO dans la clause FORMAT JSON pour diviser la valeur JSON sur plusieurs colonnes. COPY fait correspondre les attributs JSON de premier niveau avec les noms de colonnes et permet d'ingérer les valeurs imbriquées en tant que valeurs SUPER, comme les tableaux et les objets JSON.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/json/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 'auto';
```

Lorsque les noms d'attributs JSON sont en majuscules et casse mixtes, spécifiez l'option `auto ignorecase` dans la clause FORMAT JSON. Pour plus d'informations sur la commande COPY, consultez Charger des données JSON à l'aide de l'option 'auto ignorecase'.

Dans certains cas, il existe un décalage entre les noms de colonnes et les attributs JSON ou l'attribut à charger est imbriqué sur plus d'un niveau. Si c'est le cas, utilisez un fichier `jsonpaths` pour mapper manuellement les attributs JSON aux colonnes Amazon Redshift.

```
CREATE TABLE nations
(
 regionkey smallint
 ,name varchar
 ,comment super
 ,nations super
 );
```

Supposons que vous souhaitiez charger des données dans une table où les noms de colonnes ne correspondent pas aux attributs JSON. Dans l'exemple suivant, la table `nations` est une telle table. Vous pouvez créer un fichier `jsonpaths` qui fait correspondre les chemins d'accès des attributs aux colonnes de la table par leur position dans le tableau `jsonpaths`.

```
{"jsonpaths": [
       "$.r_regionkey",
       "$.r_name",
       "$.r_comment",
```

```
       "$.r_nations
    ]
}
```

L'emplacement du fichier `jsonpaths` est utilisé comme argument de FORMAT JSON.

```
COPY nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/json/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 's3://redshift-downloads/semistructured/tpch-nested/data/jsonpaths/
nations_jsonpaths.json';
```

Utilisez la requête suivante pour accéder à la table qui présente des données réparties sur plusieurs colonnes. Les colonnes de données SUPER sont imprimées en utilisant le format JSON.

```
SELECT r_regionkey,r_name,r_comment,r_nations[0].n_nationkey FROM region_nations ORDER
 BY 1,2,3 LIMIT 1;
```

Les fichiers jsonpaths mappent les champs du document JSON aux colonnes du tableau. Vous pouvez extraire des colonnes supplémentaires, telles que des clés de distribution et de tri, tout en chargeant le document complet en tant que colonne SUPER. La requête suivante charge le document complet dans la colonne nations. La colonne `name` est la clé de tri et la colonne `regionkey` est la clé de distribution.

```
CREATE TABLE nations_sorted (
    regionkey smallint,
    name varchar,
    nations super
) DISTKEY(regionkey) SORTKEY(name);
```

La racine jsonpath « $ » mappe à la racine du document comme suit :

```
{"jsonpaths": [
       "$.r_regionkey",
       "$.r_name",
       "$"
    ]
}
```

L'emplacement du fichier jsonpaths est utilisé comme argument pour FORMAT JSON.

```
COPY nations_sorted FROM 's3://redshift-downloads/semistructured/tpch-nested/data/json/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 's3://redshift-downloads/semistructured/tpch-nested/data/jsonpaths/
nations_sorted_jsonpaths.json';
```

Copie de données à partir de texte et CSV

Amazon Redshift représente des colonnes SUPER au format texte et CSV en tant que JSON sérialisés. Un formatage JSON valide est requis pour que les colonnes SUPER soient chargées avec les informations type correctes. Supprimez les objets, les tableaux, les nombres, les valeurs booléennes et les valeurs null. Placez les valeurs de chaîne entre guillemets. Les colonnes SUPER utilisent des règles d'échappement standard pour les formats texte et CSV. Pour CSV, les délimiteurs sont échappés conformément à la norme CSV. Pour le format texte, si le délimiteur choisi peut également apparaître dans un champ SUPER, utilisez l'option ESCAPE pour les instructions COPY et UNLOAD.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/csv/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT CSV;
```

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/text/
region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
DELIMITER ','
ESCAPE;
```

Copie de données à partir de Parquet et ORC au format colonne

Si vos données semi-structurées ou imbriquées sont déjà disponibles au format Apache Parquet ou Apache ORC, vous pouvez utiliser l'instruction COPY pour ingérer des données dans Amazon Redshift.

La structure de la table Amazon Redshift doit correspondre au nombre de colonnes et aux types de données de colonne des fichiers Parquet ou ORC. En spécifiant SERIALIZETOJSON dans l'instruction COPY, vous pouvez charger n'importe quel type de colonne dans le fichier qui s'aligne sur une colonne SUPER de la table en tant que SUPER. Cela inclut les types de structure et de tableau.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/
parquet/region_nation'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT PARQUET SERIALIZETOJSON;
```

L'exemple suivant utilise un format ORC.

```
COPY region_nations FROM 's3://redshift-downloads/semistructured/tpch-nested/data/orc/
region_nation'
IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT ORC SERIALIZETOJSON;
```

Lorsque les attributs des types de données de date ou d'heure sont en ORC, Amazon Redshift les convertit en varchar lors de leur encodage en SUPER.

Déchargement des données semi-structurées

Avec Amazon Redshift, vous pouvez exporter des données semi-structurées de votre cluster Amazon Redshift vers Amazon S3 dans différents formats, notamment du texte, Apache Parquet, Apache ORC et Avro. Les sections suivantes vous guideront tout au long du processus de configuration et d'exécution des opérations de déchargement pour vos données semi-structurées dans Amazon Redshift.

CSV or text formats

Vous pouvez décharger des tables contenant des colonnes de données SUPER sur Amazon S3 dans un format CSV (valeurs séparées par des virgules) ou texte. En utilisant une combinaison de clauses de navigation et de désimbrication, Amazon Redshift décharge les données hiérarchiques au format de données SUPER vers Amazon S3 au format CSV ou texte. Par la suite, vous pouvez créer des tables externes à partir des données déchargées et les interroger à l'aide de Redshift Spectrum. Pour plus d'informations sur l'utilisation de UNLOAD et les autorisations IAM requises, consultez UNLOAD.

Avant d'exécuter l'exemple suivant, renseignez la table region_nations à l'aide des processus décrits dans Chargement de données semi-structurées dans Amazon Redshift. Pour en savoir plus sur les tables utilisées dans l'exemple suivant, consultez Jeu de données échantillon SUPER.

L'exemple suivant décharge des données dans Amazon S3.

```
UNLOAD ('SELECT * FROM region_nations')
```

```
TO 's3://xxxxxx/'
IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3-Write'
DELIMITER AS '|'
GZIP
ALLOWOVERWRITE;
```

Contrairement à d'autres types de données où une chaîne définie par l'utilisateur représente une valeur nulle, Amazon Redshift exporte les colonnes de données SUPER en utilisant le format JSON et la représente comme une valeur nulle, comme déterminé par le format JSON. Par conséquent, les colonnes de données SUPER ignorent l'option NULL [AS] utilisée dans les instructions UNLOAD.

Parquet format

Vous pouvez décharger des tables comportant des colonnes de données SUPER vers Amazon S3 au format Parquet. Amazon Redshift représente les colonnes SUPER dans Parquet en tant que type de données JSON. Cela permet de représenter des données semi-structurées au format Parquet. Vous pouvez interroger ces colonnes à l'aide de Redshift Spectrum ou les intégrer dans Amazon Redshift à l'aide de la commande COPY. Pour plus d'informations sur l'utilisation de UNLOAD et les autorisations IAM requises, consultez UNLOAD.

L'exemple suivant décharge des données dans Amazon S3 au format Parquet.

```
UNLOAD ('SELECT * FROM region_nations')
TO 's3://xxxxxx/'
IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3-Write'
FORMAT PARQUET;
```

Interrogation de données semi-structurées

Avec Amazon Redshift, vous pouvez interroger et analyser des données semi-structurées, telles que JSON, Avro ou Ion, parallèlement à vos données structurées. Les données semi-structurées font référence à des données dotées d'un schéma flexible, permettant des structures hiérarchiques ou imbriquées. Les sections suivantes montrent comment interroger des données semi-structurées à l'aide de la prise en charge des formats de données ouverts par Amazon Redshift, ce qui vous permet de débloquer des informations précieuses à partir de structures de données complexes.

Amazon Redshift utilise le langage PartiQL pour offrir un accès compatible SQL aux données relationnelles, semi-structurées et imbriquées.

PartiQL fonctionne avec des types dynamiques. Cette approche permet un filtrage, une jonction et une agrégation intuitifs sur la combinaison de jeux de données structurés, semi-structurés et imbriqués. La syntaxe PartiQL utilise la notation par points et l'index de tableau pour la navigation dans les chemins lors de l'accès aux données imbriquées. Il permet également aux éléments de la clause FROM d'itérer sur des tableaux et de les utiliser pour des opérations de désimbrication. Ci-dessous, vous pouvez trouver des descriptions des différents modèles de requête qui combinent l'utilisation du type de données SUPER avec la navigation par chemin et par tableau, la désimbrication, le dépivotement ou les jointures.

Pour plus d'informations sur les tables utilisées dans l'exemple suivant, y compris le schéma, consultezJeu de données échantillon SUPER.

Rubriques

- Navigation
- Désimbriquer des requêtes
- Dépivotement d'objet
- Typage dynamique
- Sémantique laxiste
- Types d'introspection
- Classer par

Navigation

Amazon Redshift utilise PartiQL pour permettre la navigation dans les tableaux et les structures en utilisant respectivement la notation entre crochets [...] et la notation par points. En outre, vous pouvez mélanger la navigation dans des structures en utilisant notation par points avec la navigation dans des tableaux en utilisant la notation entre crochets. Par exemple, l'exemple suivant suppose que la colonne de données `c_orders` SUPER est un tableau structuré et qu'un attribut est nommé `o_orderkey`.

Pour intégrer des données dans la table `customer_orders_lineitem`, exécutez la commande suivante. Remplacez le rôle IAM par vos propres informations d'identification.

```
COPY customer_orders_lineitem FROM 's3://redshift-downloads/semistructured/tpch-nested/
data/json/customer_orders_lineitem'
REGION 'us-east-1' IAM_ROLE 'arn:aws:iam::xxxxxxxxxxxx:role/Redshift-S3'
FORMAT JSON 'auto';
```

```
SELECT c_orders[0].o_orderkey FROM customer_orders_lineitem;
```

Amazon Redshift utilise également un alias de table comme préfixe de la notation. L'exemple suivant est la même requête que celle de l'exemple précédent.

```
SELECT cust.c_orders[0].o_orderkey FROM customer_orders_lineitem AS cust;
```

Vous pouvez utiliser la notation par points et crochets dans tous les types de requêtes, comme le filtrage, la jointure et l'agrégation. Vous pouvez utiliser ces notations dans une requête dans laquelle il y a normalement des références de colonne. L'exemple suivant utilise une instruction SELECT qui filtre les résultats.

```
SELECT count(*) FROM customer_orders_lineitem WHERE c_orders[0]. o_orderkey IS NOT
 NULL;
```

L'exemple suivant utilise la notation par points et crochets dans les clauses GROUP BY et ORDER BY :

```
SELECT c_orders[0].o_orderdate,
       c_orders[0].o_orderstatus,
       count(*)
FROM customer_orders_lineitem
WHERE c_orders[0].o_orderkey IS NOT NULL
GROUP BY c_orders[0].o_orderstatus,
         c_orders[0].o_orderdate
ORDER BY c_orders[0].o_orderdate;
```

Désimbriquer des requêtes

Pour désimbriquer des requêtes, Amazon Redshift utilise la syntaxe PartiQL pour effectuer une itération sur des baies SUPER. Pour ce faire, il navigue dans le tableau à l'aide de la clause FROM d'une requête. En utilisant l'exemple précédent, le suivant itère sur les valeurs de l'attribut pour `c_orders`.

```
SELECT c.*, o FROM customer_orders_lineitem c, c.c_orders o;
```

La syntaxe de désimbrication est une extension de la clause FROM. Dans SQL standard, la clause FROM `x (AS) y` signifie que y itère sur chaque tuple dans la relation x. Dans ce cas, x fait

référence à une relation et y fait référence à un alias pour la relation x. De même, la syntaxe PartiQL de désimbrication à l'aide de l'élément de clause FROM x (AS) y signifie quey effectue une itération sur chaque valeur (SUPER) de l'expression x du tableau (SUPER). Dans ce cas, x est une expression SUPER et y est un alias pour x.

L'opérande de gauche peut également utiliser la notation par points et crochets pour la navigation régulière. Dans l'exemple précédent, customer_orders_lineitem c est l'itération sur la table de base customer_order_lineitem et c.c_orders o est l'itération sur le tableau c.c_orders. Pour itérer sur l'attribut o_lineitems, qui est un tableau dans un tableau, vous ajoutez plusieurs clauses.

```
SELECT c.*, o, l FROM customer_orders_lineitem c, c.c_orders o, o.o_lineitems l;
```

Amazon Redshift prend également en charge un index de tableau lors de l'itération sur le tableau à l'aide du mot clé AT. La clause x AS y AT z itère sur le tableau x et génère le champ z, qui est l'index du tableau. L'exemple suivant illustre le fonctionnement d'un index de tableau.

```
SELECT c_name,
       orders.o_orderkey AS orderkey,
       index AS orderkey_index
FROM customer_orders_lineitem c, c.c_orders AS orders AT index
ORDER BY orderkey_index;

c_name             | orderkey | orderkey_index
-------------------+----------+----------------
Customer#000008251 | 3020007  |        0
Customer#000009452 | 4043971  |        0
  (2 rows)
```

L'exemple suivant itère sur un tableau scalaire

```
CREATE TABLE bar AS SELECT json_parse('{"scalar_array": [1, 2.3, 45000000]}') AS data;

SELECT index, element FROM bar AS b, b.data.scalar_array AS element AT index;

 index | element
-------+----------
     0 | 1
     1 | 2.3
     2 | 45000000
```

```
(3 rows)
```

L'exemple suivant itère sur un tableau de plusieurs niveaux. L'exemple utilise plusieurs clauses de désimbrication (unnest) pour effectuer une itération dans les tableaux les plus intérieurs. Le tableau AS `f.multi_level_array` itère sur `multi_level_array`. L'élément AS du tableau est l'itération sur les tableaux dans `multi_level_array`.

```
CREATE TABLE foo AS SELECT json_parse('[[1.1, 1.2], [2.1, 2.2], [3.1, 3.2]]') AS
 multi_level_array;

SELECT array, element FROM foo AS f, f.multi_level_array AS array, array AS element;

   array   | element
-----------+---------
 [1.1,1.2] | 1.1
 [1.1,1.2] | 1.2
 [2.1,2.2] | 2.1
 [2.1,2.2] | 2.2
 [3.1,3.2] | 3.1
 [3.1,3.2] | 3.2
(6 rows)
```

Pour plus d'informations sur la clause FROM, consultez Clause FROM.

Dépivotement d'objet

Pour effectuer le dépivotement des objets, Amazon Redshift utilise la syntaxe PartiQL pour effectuer une itération sur des objets SUPER. Pour ce faire, il utilise la clause FROM d'une requête avec le mot clé UNPIVOT. Dans ce cas, l'expression est l'`c.c_orders[0]`objet. L'exemple de requête itère sur chaque attribut renvoyé par l'objet.

```
SELECT attr as attribute_name, json_typeof(val) as value_type
FROM customer_orders_lineitem c, UNPIVOT c.c_orders[0] AS val AT attr
WHERE c_custkey = 9451;

 attribute_name  | value_type
-----------------+------------
 o_orderstatus   | string
 o_clerk         | string
 o_lineitems     | array
 o_orderdate     | string
```

```
 o_shippriority  | number
 o_totalprice    | number
 o_orderkey      | number
 o_comment       | string
 o_orderpriority | string
(9 rows)
```

Comme pour la désimbrication, la syntaxe de dépivotement est une extension de la clause FROM. La différence réside dans le fait que la syntaxe de dépivotement utilise le mot clé UNPIVOT pour indiquer qu'il effectue une itération sur un objet au lieu d'un tableau. Il utilise la `value_alias` AS pour l'itération sur toutes les valeurs à l'intérieur d'un objet et utilise l'`attribute_alias` AT pour effectuer une itération sur tous les attributs. Examinez le fragment de syntaxe suivant :

```
UNPIVOT expression AS value_alias [ AT attribute_alias ]
```

Amazon Redshift prend en charge l'utilisation du dépivotement des objets et de la désimbrication des tableaux dans une seule clause FROM, comme suit :

```
SELECT attr as attribute_name, val as object_value
FROM customer_orders_lineitem c, c.c_orders AS o, UNPIVOT o AS val AT attr
WHERE c_custkey = 9451;
```

Lorsque vous utilisez le dépivotement d'objet, Amazon Redshift ne prend pas en charge le dépivotement corrélé. Concrètement, supposons que vous ayez un cas où il existe plusieurs exemples de dépivotement dans différents niveaux de requête et que le dépivotement interne fait référence à l'externe. Amazon Redshift ne prend pas en charge ce type de dépivotement multiple.

Pour plus d'informations sur la clause FROM, consultez Clause FROM. Pour bénéficier d'exemples montrant comment interroger des données structurées avec PIVOT et UNPIVOT, consultez Exemples PIVOT et UNPIVOT.

Typage dynamique

Le typage dynamique ne nécessite pas de moulage explicite des données qui sont extraites des chemins en notation points et crochets. Amazon Redshift utilise le typage dynamique pour traiter des données SUPER sans schéma sans avoir à déclarer les types de données avant de les utiliser dans votre requête. Le typage dynamique utilise les résultats de la navigation dans les colonnes de données SUPER sans devoir les convertir explicitement en types Amazon Redshift. Le typage dynamique est le plus utile dans les jointures et les clauses GROUP BY. L'exemple suivant utilise

une instruction SELECT qui ne nécessite aucune conversion explicite des expressions points et crochets aux types habituels d'Amazon Redshift. Pour plus d'informations sur la compatibilité et la conversion des types, consultez Compatibilité et conversion de types.

```
SELECT c_orders[0].o_orderkey
FROM customer_orders_lineitem
WHERE c_orders[0].o_orderstatus = 'P';
```

Le signe d'égalité dans cette requête est évalué à `true` lorsque c_orders[0].o_orderstatus est la chaîne « P ». Dans tous les autres cas, le signe d'égalité est évalué à `false`, y compris lorsque les arguments de l'égalité sont de types différents.

Typage dynamique et statique

Sans utiliser le typage dynamique, vous ne pouvez pas déterminer si c_orders[0].o_orderstatus est une chaîne, un entier ou une structure. Vous pouvez seulement déterminer que c_orders[0].o_orderstatus est un type de données SUPER, qui peut être un scalaire Amazon Redshift, un tableau ou une structure. Le type statique de c_orders[0].o_orderstatus est un type de données SUPER. Conventionnellement, un type est implicitement statique dans SQL.

Amazon Redshift utilise le typage dynamique pour le traitement des données sans schéma. Lorsque la requête évalue les données, c_orders[0].o_orderstatus s'avère être un type spécifique. Par exemple, l'évaluation de c_orders[0].o_orderstatus sur le premier enregistrement de customer_orders_lineitem peut aboutir à un entier. L'évaluation sur le deuxième enregistrement peut résulter en une chaîne de caractères. Ce sont les types dynamiques de l'expression.

Lors de l'utilisation d'un opérateur ou d'une fonction SQL avec des expressions de type point et crochets ayant des types dynamiques, Amazon Redshift produit des résultats similaires à l'utilisation d'un opérateur ou d'une fonction SQL standard avec les types statiques respectifs. Dans cet exemple, lorsque le type dynamique de l'expression de chemin est une chaîne, la comparaison avec la chaîne « P » est significative. Si le type dynamique de c_orders[0].o_orderstatus est d'un autre type de données que celui de chaîne de caractères, l'égalité renvoie faux. Les autres fonctions renvoient null lorsque des arguments mal typés sont utilisés.

L'exemple suivant écrit la requête précédente avec un typage statique :

```
SELECT c_custkey
FROM customer_orders_lineitem
WHERE CASE WHEN JSON_TYPEOF(c_orders[0].o_orderstatus) = 'string'
           THEN c_orders[0].o_orderstatus::VARCHAR = 'P'
```

```
            ELSE FALSE END;
```

Notez la distinction suivante entre les prédicats d'égalité et les prédicats de comparaison. Dans l'exemple précédent, si vous remplacez le prédicat d'égalité par un less-than-or-equal prédicat, la sémantique produit une valeur nulle au lieu de fausse.

```
SELECT c_orders[0]. o_orderkey
FROM customer_orders_lineitem
WHERE c_orders[0].o_orderstatus <= 'P';
```

Dans cet exemple, si c_orders[0].o_orderstatus est une chaîne, Amazon Redshift renvoie true si elle est alphabétiquement égale ou inférieure à « P ». Amazon Redshift renvoie false si elle est alphabétiquement supérieure à « P ». Toutefois, si c_orders[0].o_orderstatus n'est pas une chaîne, Amazon Redshift renvoie null car Amazon Redshift ne peut pas comparer des valeurs de différents types, comme indiqué dans la requête suivante :

```
SELECT c_custkey
FROM customer_orders_lineitem
WHERE CASE WHEN JSON_TYPEOF(c_orders[0].o_orderstatus) = 'string'
           THEN c_orders[0].o_orderstatus::VARCHAR <= 'P'
           ELSE NULL END;
```

Le typage dynamique n'exclut pas des comparaisons des types qui sont minimalement comparables. Par exemple, vous pouvez convertir les types scalaires CHAR et VARCHAR Amazon Redshift en SUPER. Ils sont comparables à des chaînes de caractères, y compris en ignorant les caractères d'espacement de fin de chaîne, comme pour les types CHAR et VARCHAR d'Amazon Redshift. De même, les valeurs entières, décimales et à virgule flottante sont comparables en tant que valeurs SUPER. Spécifiquement pour les colonnes décimales, chaque valeur peut également avoir une graduation différente. Amazon Redshift les considère quand même comme des types dynamiques.

Amazon Redshift prend également en charge l'égalité sur les objets et les tableaux évalués comme étant profondément égaux, comme l'évaluation profonde des objets ou des tableaux et la comparaison de tous les attributs. Utilisez l'égalité profonde avec prudence, car le processus d'exécution de l'égalité profonde peut prendre du temps.

Utilisation du typage dynamique pour les jointures

Pour les jointures, le typage dynamique fait automatiquement correspondre des valeurs avec différents types dynamiques sans avoir à effectuer une longue analyse CASE WHEN pour savoir

quels types de données peuvent apparaître. Supposons par exemple que votre organisation ait changé le format qu'elle utilisait pour les clés partielles (part keys) au fil du temps.

Les clés partielles de type entier initialement émises sont remplacées par des clés partielles de type chaîne de caractères, telles que 'A55', puis à nouveau par des clés partielles de type tableau, telles que ['X', 10] combinant une chaîne de caractères et un nombre. Amazon Redshift n'a pas besoin d'effectuer une longue analyse de cas sur les clés partielles et peut utiliser des jointures comme indiqué dans l'exemple suivant.

```
SELECT c.c_name
    ,l.l_extendedprice
    ,l.l_discount
FROM customer_orders_lineitem c
    ,c.c_orders o
    ,o.o_lineitems l
    ,supplier_partsupp s
    ,s.s_partsupps ps
WHERE l.l_partkey = ps.ps_partkey
AND c.c_nationkey = s.s_nationkey
ORDER BY c.c_name;
```

L'exemple suivant montre à quel point la même requête peut être complexe et inefficace sans utiliser le typage dynamique :

```
SELECT c.c_name
    ,l.l_extendedprice
    ,l.l_discount
FROM customer_orders_lineitem c
    ,c.c_orders o
    ,o.o_lineitems l
    ,supplier_partsupp s
    ,s.s_partsupps ps
WHERE CASE WHEN IS_INTEGER(l.l_partkey) AND IS_INTEGER(ps.ps_partkey)
           THEN l.l_partkey::integer = ps.ps_partkey::integer
           WHEN IS_VARCHAR(l.l_partkey) AND IS_VARCHAR(ps.ps_partkey)
           THEN l.l_partkey::varchar = ps.ps_partkey::varchar
           WHEN IS_ARRAY(l.l_partkey) AND IS_ARRAY(ps.ps_partkey)
                AND IS_VARCHAR(l.l_partkey[0]) AND IS_VARCHAR(ps.ps_partkey[0])
                AND IS_INTEGER(l.l_partkey[1]) AND IS_INTEGER(ps.ps_partkey[1])
           THEN l.l_partkey[0]::varchar = ps.ps_partkey[0]::varchar
                AND l.l_partkey[1]::integer = ps.ps_partkey[1]::integer
           ELSE FALSE END
```

```
AND c.c_nationkey = s.s_nationkey
ORDER BY c.c_name;
```

Sémantique laxiste

Par défaut, les opérations de navigation sur les valeurs SUPER renvoient null au lieu de renvoyer une erreur lorsque la navigation n'est pas valide. La navigation par objet est invalide si la valeur SUPER n'est pas un objet ou si la valeur SUPER est un objet mais ne contient pas le nom de l'attribut utilisé dans la requête. Par exemple, la requête suivante accède à un nom d'attribut non valide dans la colonne de données SUPER cdata :

```
SELECT c.c_orders.something FROM customer_orders_lineitem c;
```

La navigation de tableau renvoie null si la valeur SUPER n'est pas un tableau ou si l'index du tableau est hors limites. La requête suivante renvoie null car c_orders[1][1] est hors limites.

```
SELECT c.c_orders[1][1] FROM customer_orders_lineitem c;
```

La sémantique laxiste est particulièrement utile lorsqu'on utilise le typage dynamique pour convertir une valeur SUPER. Le transtypage d'une valeur SUPER en un type incorrect renvoie null au lieu d'une erreur si la conversion n'est pas valide. Par exemple, la requête suivante renvoie null car elle ne peut pas convertir la valeur de chaîne « Good » de l'attribut d'objet o_orderstatus en INTEGER. Amazon Redshift renvoie une erreur pour une conversion de VARCHAR en INTEGER mais pas pour une conversion en SUPER.

```
SELECT c.c_orders.o_orderstatus::integer FROM customer_orders_lineitem c;
```

Types d'introspection

Les colonnes de données SUPER prennent en charge les fonctions d'inspection qui renvoient le type dynamique et d'autres informations de type sur la valeur SUPER. L'exemple le plus courant est la fonction scalaire JSON_TYPEOF qui renvoie un VARCHAR avec les valeurs booléen (boolean), nombre (number), chaîne (string), objet (object), tableau (array) ou null, selon le type dynamique de la valeur SUPER. Amazon Redshift prend en charge les fonctions booléennes suivantes pour les colonnes de données SUPER :

- DECIMAL_PRECISION

- DECIMAL_SCALE
- IS_ARRAY
- IS_BIGINT
- IS_CHAR
- IS_DECIMAL
- IS_FLOAT
- IS_INTEGER
- IS_OBJECT
- IS_SCALAR
- IS_SMALLINT
- IS_VARCHAR
- JSON_TYPEOF

Toutes ces fonctions renvoient false si la valeur d'entrée est nulle. IS_SCALAR, IS_OBJECT et IS_ARRAY s'excluent mutuellement et couvrent toutes les valeurs possibles à l'exception de null.

Pour déduire les types correspondant aux données, Amazon Redshift utilise la fonction JSON_TYPEOF qui renvoie le type de (premier niveau de) la valeur SUPER comme indiqué dans l'exemple suivant :

```
SELECT JSON_TYPEOF(r_nations) FROM region_nations;
 json_typeof
 -------------
 array
(1 row)
```

```
SELECT JSON_TYPEOF(r_nations[0].n_nationkey) FROM region_nations;
 json_typeof
 -------------
 number
```

Amazon Redshift voit cela comme une longue chaîne unique, ce qui revient à insérer cette valeur dans une colonne VARCHAR au lieu d'un SUPER. Puisque la colonne est SUPER, la chaîne unique est toujours une valeur SUPER valide et la différence est notée dans JSON_TYPEOF :

```
SELECT IS_VARCHAR(r_nations[0].n_name) FROM region_nations;
 is_varchar
-------------
 true
(1 row)
```

```
SELECT r_nations[4].n_name FROM region_nations
WHERE CASE WHEN IS_INTEGER(r_nations[4].n_nationkey)
           THEN r_nations[4].n_nationkey::INTEGER = 15
           ELSE false END;
```

Classer par

Amazon Redshift ne définit pas de comparaisons SUPER entre des valeurs ayant des types dynamiques différents. Une valeur SUPER qui est une chaîne n'est ni plus petite ni plus grande qu'une valeur SUPER qui est un nombre. Pour utiliser les clauses ORDER BY avec les colonnes SUPER, Amazon Redshift définit un ordre total parmi les différents types à observer lorsque Amazon Redshift classe les valeurs SUPER à l'aide des clauses ORDER BY. L'ordre des types dynamiques est le suivant : booléen, nombre, chaîne, tableau, objet. L'exemple suivant montre les différents types d'ordres :

```
INSERT INTO region_nations VALUES
(100,'name1','comment1','AWS'),
(200,'name2','comment2',1),
(300,'name3','comment3',ARRAY(1, 'abc', null)),
(400,'name4','comment4',-2.5),
(500,'name5','comment5','Amazon');

SELECT r_nations FROM region_nations order by r_nations;

r_nations
----------------
 -2.5
 1
 "Amazon"
 "AWS"
 [1,"abc",null]
(5 rows)
```

Pour plus d'informations sur la clause ORDER BY, consultez Clause ORDER BY.

Opérateurs et fonctions

Avec Amazon Redshift, vous pouvez effectuer des analyses avancées sur de grands ensembles de données à l'aide de données SUPER à l'aide d'opérateurs et de fonctions. Les opérateurs et les fonctions des données SUPER sont des constructions SQL qui permettent une analyse et une manipulation complexes de données semi-structurées stockées dans des tables Amazon Redshift.

Les sections suivantes présentent la syntaxe, les exemples et les meilleures pratiques d'utilisation des opérateurs et des fonctions pour les données SUPER dans Amazon Redshift afin d'exploiter tout le potentiel de vos données semi-structurées.

Opérateurs arithmétiques

Les valeurs SUPER prennent en charge tous les opérateurs arithmétiques de base +, -, *, /, % en utilisant le typage dynamique. Le type résultant de l'opération reste SUPER. Pour tous les opérateurs, à l'exception de l'opérateur binaire +, les opérandes d'entrée doivent être des nombres. Sinon, Amazon Redshift renvoie null. La distinction entre les valeurs décimales et les valeurs à virgule flottante est conservée lorsque Amazon Redshift exécute ces opérateurs et que le type dynamique ne change pas. Cependant, l'échelle décimale change lorsque vous utilisez des multiplications et des divisions. Les débordements arithmétiques provoquent toujours des erreurs de requête, ils ne sont pas modifiés en null. L'opérateur binaire + effectue une addition si les entrées sont des nombres ou une concaténation si les entrées sont des chaînes de caractères. Si un opérande est une chaîne et que l'autre opérande est un nombre, le résultat est nul. Les opérateurs préfixes unaires + et - renvoient un résultat null si la valeur SUPER n'est pas un nombre, comme le montre l'exemple suivant :

```
SELECT (c_orders[0]. o_orderkey + 0.5) * c_orders[0]. o_orderkey / 10 AS math FROM
 customer_orders_lineitem;
            math
----------------------------
 1757958232200.1500
(1 row)
```

Le typage dynamique permet aux valeurs décimales de SUPER d'avoir des échelles différentes. Amazon Redshift traite les valeurs décimales comme s'il s'agissait de types statiques différents et autorise toutes les opérations mathématiques. Amazon Redshift calcule l'échelle résultante dynamiquement en fonction des échelles des opérandes. Si l'un des opérandes est un nombre à virgule flottante, Amazon Redshift promeut l'autre opérande à un nombre à virgule flottante et génère le résultat sous la forme d'un nombre à virgule flottante.

Fonctions arithmétiques

Amazon Redshift prend en charge les fonctions arithmétiques suivantes pour les colonnes SUPER. Ils retournent null si l'entrée n'est pas un nombre :

- FLOOR. Pour de plus amples informations, veuillez consulter Fonction FLOOR.
- CEIL et CEILING. Pour de plus amples informations, veuillez consulter Fonction CEILING (ou CEIL).
- ROUND. Pour de plus amples informations, veuillez consulter Fonction ROUND.
- TRUNC. Pour de plus amples informations, veuillez consulter Fonction TRUNC.
- ABS. Pour de plus amples informations, veuillez consulter Fonction ABS.

L'exemple suivant utilise des fonctions arithmétiques pour interroger des données :

```
SELECT x, FLOOR(x), CEIL(x), ROUND(x)
FROM (
    SELECT (c_orders[0]. o_orderkey + 0.5) * c_orders[0].o_orderkey / 10 AS x
    FROM customer_orders_lineitem
    );

         x          |     floor     |     ceil      |     round
--------------------+---------------+---------------+---------------
 1389636795898.0500  | 1389636795898  | 1389636795899  | 1389636795898
```

La fonction ABS conserve l'échelle de la décimale d'entrée contrairement aux fonctions FLOOR, CEIL. ROUND élimine l'échelle de la décimale en entrée.

Fonctions de tableau

Amazon Redshift prend en charge la composition de tableaux et les fonctions utilitaires suivantes : array, array_concat, subarray, array_flatten, get_array_length et split_to_array.

Vous pouvez construire des tableaux SUPER à partir de valeurs de types de données Amazon Redshift en utilisant la fonction ARRAY, y compris d'autres valeurs SUPER. L'exemple suivant utilise la fonction variadique ARRAY :

```
SELECT ARRAY(1, c.c_custkey, NULL, c.c_name, 'abc') FROM customer_orders_lineitem c;
```

```
                                array
 ------------------------------------------------------
[1,8401,null,""Customer#000008401"",""abc""]
[1,9452,null,""Customer#000009452"",""abc""]
[1,9451,null,""Customer#000009451"",""abc""]
[1,8251,null,""Customer#000008251"",""abc""]
[1,5851,null,""Customer#000005851"",""abc""]
(5 rows)
```

L'exemple suivant utilise la concaténation de tableau avec la fonction ARRAY_CONCAT :

```
SELECT ARRAY_CONCAT(JSON_PARSE('[10001,10002]'),JSON_PARSE('[10003,10004]'));

             array_concat
-----------------------------------
 [10001,10002,10003,10004]
(1 row)
```

L'exemple suivant utilise la manipulation de tableau avec la fonction SUBARRAY qui renvoie un sous-ensemble du tableau d'entrée.

```
SELECT SUBARRAY(ARRAY('a', 'b', 'c', 'd', 'e', 'f'), 2, 3);

   subarray
---------------
 ["c","d","e"]
(1 row))
```

L'exemple suivant fusionne plusieurs niveaux de tableaux en un seul tableau en utilisant ARRAY_FLATTEN :

```
SELECT x, ARRAY_FLATTEN(x) FROM (SELECT ARRAY(1, ARRAY(2, ARRAY(3, ARRAY())))  AS x);

     x           | array_flatten
 ----------------+---------------
 [1,[2,[3,[]]]]  | [1,2,3]
(1 row)
```

Les fonctions de tableau ARRAY_CONCAT et ARRAY_FLATTEN utilisent des règles de typage dynamique. Elles retournent une valeur null au lieu d'une erreur si l'entrée n'est pas un tableau. La

fonction GET_ARRAY_LENGTH renvoie la longueur d'un SUPER tableau à partir du chemin d'un objet ou d'un tableau.

```
SELECT c_name
FROM customer_orders_lineitem
WHERE GET_ARRAY_LENGTH(c_orders) = (
    SELECT MAX(GET_ARRAY_LENGTH(c_orders))
    FROM customer_orders_lineitem
    );
```

L'exemple suivant divise une chaîne en un tableau de chaînes en utilisant SPLIT_TO_ARRAY. La fonction utilise un délimiteur comme paramètre facultatif. Si aucun délimiteur n'est défini, la valeur par défaut est une virgule.

```
SELECT SPLIT_TO_ARRAY('12|345|6789', '|');

   split_to_array
---------------------
 ["12","345","6789"]
(1 row)
```

Configurations SUPER

Vous pouvez configurer vos données SUPER pour des scénarios spécifiques. Les sections suivantes fournissent des détails sur le choix et l'application des configurations SUPER appropriées en fonction de vos exigences en matière de format de données.

Rubriques

- Modes laxistes et stricts pour SUPER
- Accès aux champs JSON avec des noms de champs ou des attributs en majuscules et à casse mixte
- Options d'analyse pour Super

Modes laxistes et stricts pour SUPER

Lorsque vous interrogez des données SUPER, l'expression de chemin peut ne pas correspondre à la structure de données SUPER réelle. Si vous essayez d'accéder à un membre inexistant d'un objet ou à un élément d'un tableau, Amazon Redshift renvoie une valeur NULL si votre requête est exécutée

dans le mode lax par défaut. Si vous exécutez votre requête en mode strict, Amazon Redshift renvoie une erreur. Les paramètres de session suivants peuvent être définis pour activer ou désactiver le mode lax.

L'exemple suivant utilise des paramètres de session pour activer le mode lax.

```
SET navigate_super_null_on_error=ON;  --default lax mode for navigation

SET cast_super_null_on_error=ON;  --default lax mode for casting

SET parse_super_null_on_error=OFF;  --default strict mode for ingestion
```

Accès aux champs JSON avec des noms de champs ou des attributs en majuscules et à casse mixte

Lorsque les noms de vos attributs JSON sont en majuscules ou à casse mixte, vous devez être en mesure de naviguer dans les structures de type SUPER en tenant compte de la casse. Pour ce faire, vous pouvez configurer `enable_case_sensitive_identifier` sur TRUE et entourer les noms d'attributs en majuscules et à casse mixte de guillemets doubles. Vous pouvez également configurer `enable_case_sensitive_super_attribute` sur TRUE. Dans ce cas, vous pouvez utiliser les noms d'attributs en majuscules et à casse mixte dans vos requêtes sans les mettre entre guillemets.

L'exemple suivant illustre comment configurer `enable_case_sensitive_identifier` pour interroger des données.

```
SET enable_case_sensitive_identifier to TRUE;

-- Accessing JSON attribute names with uppercase and mixedcase names
SELECT json_table.data."ITEMS"."Name",
       json_table.data."price"
FROM
  (SELECT json_parse('{"ITEMS":{"Name":"TV"}, "price": 345}') AS data) AS json_table;

 Name | price
------+-------
 "TV" | 345
(1 row)

RESET enable_case_sensitive_identifier;
```

```
-- After resetting the above configuration, the following query accessing JSON
 attribute names with uppercase and mixedcase names should return null (if in lax
 mode).
SELECT json_table.data."ITEMS"."Name",
       json_table.data."price"
FROM
  (SELECT json_parse('{"ITEMS":{"Name":"TV"}, "price": 345}') AS data) AS json_table;

 name | price
------+-------
      | 345
(1 row)
```

L'exemple suivant illustre comment configurer `enable_case_sensitive_super_attribute` pour interroger des données.

```
SET enable_case_sensitive_super_attribute to TRUE;
 -- Accessing JSON attribute names with uppercase and mixedcase names

SELECT json_table.data.ITEMS.Name,
       json_table.data.price
FROM
  (SELECT json_parse('{"ITEMS":{"Name":"TV"}, "price": 345}') AS data) AS json_table;

 name | price
 -----+-------
 "TV" | 345
(1 row)

 RESET enable_case_sensitive_super_attribute;

 -- After resetting enable_case_sensitive_super_attribute, the query now returns NULL
 for ITEMS.Name (if in lax mode).

SELECT json_table.data.ITEMS.Name,
       json_table.data.price
FROM
  (SELECT json_parse('{"ITEMS":{"Name":"TV"}, "price": 345}') AS data) AS json_table;

 name | price
 -----+-------
      | 345
(1 row)
```

Options d’analyse pour Super

Lorsque vous utilisez la fonction JSON_PARSE pour analyser des chaînes JSON en valeurs SUPER, certaines restrictions s’appliquent :

- Le même nom d’attribut ne peut pas apparaître dans le même objet, mais peut apparaître dans un objet imbriqué. L’option de configuration `json_parse_dedup_attributes` permet à JSON_PARSE de ne conserver que la dernière occurrence d’attributs dupliqués au lieu de renvoyer une erreur.
- Les valeurs de chaîne ne peuvent pas dépasser la taille varchar maximale du système de 65 535 octets. L’option de configuration `json_parse_truncate_strings` permet à JSON_PARSE() de tronquer automatiquement des chaînes qui dépassent cette limite sans renvoyer d’erreur. Ce comportement affecte uniquement les valeurs de chaîne et non les noms d’attribut.

Pour plus d’informations sur la fonction JSON_PARSE, consultez Fonction JSON_PARSE.

L’exemple suivant illustre comment définir l’option de configuration `json_parse_dedup_attributes` pour le comportement par défaut de renvoi d’une erreur en cas d’attributs en double.

```
SET json_parse_dedup_attributes=OFF;  --default behavior of returning error instead of
 de-duplicating attributes
```

L’exemple suivant illustre comment définir l’option de configuration `json_parse_truncate_strings` pour le comportement par défaut de renvoi d’une erreur en cas de chaînes de longueur supérieure à cette limite.

```
SET json_parse_truncate_strings=OFF;  --default behavior of returning error instead of
 truncating strings
```

Limites

Avec Amazon Redshift, vous pouvez utiliser le type de données SUPER pour stocker et interroger des données semi-structurées telles que JSON, Avro ou Ion. Les limites du type de données SUPER font référence aux contraintes et aux limites associées à l'utilisation de ce type de données dans Amazon Redshift. Les sections suivantes fournissent des détails sur les limites spécifiques du type de

données SUPER, telles que la taille maximale, les niveaux d'imbrication et les types de données pris en charge dans les données semi-structurées.

- Vous ne pouvez pas définir les colonnes SUPER en tant que clé de distribution ou de tri.
- Un objet SUPER individuel peut contenir jusqu'à 16 Mo de données.
- La profondeur d'imbrication maximale pour les tableaux et les structures du type de données SUPER est de 1 000.
- Une valeur individuelle dans un objet SUPER est limitée à la longueur maximale du type Amazon Redshift correspondant. Par exemple, une valeur de chaîne unique chargée sur SUPER est limitée à la longueur VARCHAR maximale de 65 535 octets.
- Vous ne pouvez pas effectuer d'opérations de mise à jour partielle ou de transformation sur les colonnes SUPER.
- Vous ne pouvez pas utiliser le type de données SUPER et ses alias dans des jointures droites ou des jointures externes complètes.
- Le type de données SUPER ne prend pas en charge XML en tant que format de sérialisation entrant ou sortant.
- Dans la clause FROM d'une sous-requête (corrélée ou non) faisant référence à une variable de table pour la désimbrication, la requête ne peut faire référence qu'à sa table parente et non à d'autres tables.
- Vous ne pouvez pas effectuer de dénidification ou de dépivotement d'objet sur des colonnes SUPER produites par une expression de table commune récursive (). `WITH RECURSIVE` Pour plus d'informations sur la récursivité CTEs, consultezExpressions récursives de table commune.
- Limites de transtypage (casting)

  Les valeurs SUPER peuvent être converties vers et depuis d'autres types de données, avec les exceptions suivantes :

  - Amazon Redshift ne différencie pas les entiers et les décimales d'échelle 0.
  - Si l'échelle est différente de zéro, le type de données SUPER a le même comportement que les autres types de données Amazon Redshift, si ce n'est qu'Amazon Redshift convertit les erreurs liées à SUPER en null, comme le montre l'exemple suivant.

    ```
    SELECT 5::bool;
     bool
    -------
     True
    (1 row)
    ```

```
SELECT 5::decimal::bool;
ERROR:  cannot cast type numeric to boolean

SELECT 5::super::bool;
 bool
-------
 True
(1 row)

SELECT 5.0::bool;
ERROR:  cannot cast type numeric to boolean

SELECT 5.0::super::bool;
 bool
-------
(1 row)
```

- Amazon Redshift ne convertit pas les types de date et d'heure en type de données SUPER. Amazon Redshift ne peut que convertir les types de données date et heure à partir du type de données SUPER, comme le montre l'exemple suivant.

```
SELECT o.o_orderdate FROM customer_orders_lineitem c,c.c_orders o;
  order_date
------------------
 "2001-09-08"
(1 row)


SELECT JSON_TYPEOF(o.o_orderdate) FROM customer_orders_lineitem c,c.c_orders o;
 json_typeof
-----------------
 string
(1 row)


SELECT o.o_orderdate::date FROM customer_orders_lineitem c,c.c_orders o;
 order_date
----------------
 2001-09-08
(1 row)
```

```
--date/time cannot be cast to super
SELECT '2019-09-09'::date::super;
ERROR:  cannot cast type date to super
```

- La conversion de valeurs non scalaires (objets et tableaux) en chaînes de caractères renvoie NULL. Pour sérialiser correctement ces valeurs non scalaires, ne les convertissez pas. Utilisez plutôt `json_serialize` pour convertir les valeurs non scalaires. La fonction `json_serialize` renvoie un varchar. En règle générale, vous n'avez pas besoin de convertir les valeurs non scalaires en varchar car Amazon Redshift sérialise implicitement, comme le montre le premier exemple suivant.

```
SELECT r_nations FROM region_nations WHERE r_regionkey=300;
   r_nations
----------------
 [1,"abc",null]
(1 row)

SELECT r_nations::varchar FROM region_nations WHERE r_regionkey=300;
 r_nations
-------------
(1 row)

SELECT JSON_SERIALIZE(r_nations) FROM region_nations WHERE r_regionkey=300;
 json_serialize
-----------------
 [1,"abc",null]
(1 row)
```

- Pour les bases de données non sensibles à la casse, Amazon Redshift ne prend pas en charge le type de données SUPER. Pour les colonnes non sensibles à la casse, Amazon Redshift ne les transforme pas en type SUPER. Par conséquent, Amazon Redshift ne prend pas en charge les colonnes SUPER interagissant avec les colonnes non sensibles à la casse qui déclenchent la conversion.

- Amazon Redshift ne prend pas en charge les fonctions volatiles, telles que RANDOM ( ) ou TIMEOFDAY ( ), dans les sous-requêtes qui désintègrent une table externe ou un côté gauche (LHS) de fonctions IN avec de telles sous-requêtes.

Type de données SUPER et vues matérialisées

Avec Amazon Redshift, vous pouvez utiliser le type de données SUPER pour améliorer les performances et la flexibilité des vues matérialisées. Le type de données SUPER vous permet de stocker un sur-ensemble de colonnes provenant des tables de base dans une vue matérialisée, ce qui vous permet d'interroger directement la vue matérialisée sans joindre les tables de base. Les sections suivantes expliquent comment créer et utiliser des vues matérialisées avec le type de données SUPER dans Amazon Redshift.

Amazon Redshift étend la capacité des vues matérialisées pour travailler avec le type de données SUPER et PartiQL dans les vues matérialisées. Les requêtes SQL et PartiQL peuvent être précalculées à l'aide de vues matérialisées incrémentielles. Pour plus d'informations sur les vues matérialisées, consultez Vues matérialisées dans Amazon Redshift.

Une fois que vous avez stocké vos données sans schéma et semi-structurées dans SUPER, vous pouvez utiliser les vues matérialisées PartiQL pour introspecter les données et les fragmenter en vues matérialisées.

Accélération des requêtes PartiQL

Vous pouvez utiliser les vues matérialisées pour accélérer les requêtes PartiQL qui parcourent et/ou désimbriquent les données hiérarchiques dans les colonnes SUPER. Créez une ou plusieurs vues matérialisées pour fragmenter les valeurs SUPER en plusieurs colonnes et utiliser l'organisation en colonnes des requêtes analytiques Amazon Redshift. Par conséquent, les requêtes font appel aux vues matérialisées.

La vue matérialisée extrait et normalise essentiellement les données imbriquées. Le niveau de normalisation dépend de l'effort que vous déployez pour transformer les données SUPER en données colonnaires conventionnelles.

Fragmentation en colonnes SUPER avec des vues matérialisées

Avec Amazon Redshift, vous pouvez améliorer les performances des requêtes et réduire les besoins en stockage en répartissant les données dans des colonnes SUPER à l'aide de vues matérialisées. Le déchiquetage fait référence au processus qui consiste à décomposer des types de données complexes tels que le JSON semi-structuré ou le XML en colonnes plus petites et plus plates. Les colonnes SUPER sont une forme spécialisée de stockage en colonnes optimisée pour analyser rapidement les données déchiquetées.

Les sections suivantes décrivent les étapes et les considérations relatives au broyage de données dans des colonnes SUPER à l'aide de vues matérialisées dans Amazon Redshift.

L'exemple suivant montre une vue matérialisée qui fragmente les données imbriquées, les colonnes résultantes étant toujours du type de données SUPER.

```
SELECT c.c_name, o.o_orderstatus
FROM customer_orders_lineitem c, c.c_orders o;
```

L'exemple suivant illustre une vue matérialisée qui crée des colonnes scalaires Amazon Redshift conventionnelles à partir des données fragmentées.

```
SELECT c.c_name, c.c_orders[0].o_totalprice
FROM customer_orders_lineitem c;
```

Vous pouvez créer une vue matérialisée unique super_mv pour accélérer les deux requêtes.

Pour répondre à la première requête, vous devez matérialiser l'attribut o_orderstatus. Vous pouvez omettre l'attribut c_name car il n'implique pas la navigation imbriquée ni la désimbrication. Vous devez également inclure dans la vue matérialisée l'attribut c_custkey de customer_orders_lineitem pour pouvoir joindre la table de base à la vue matérialisée.

Pour répondre à la deuxième requête, vous devez également matérialiser l'attribut o_totalprice et l'index de tableau o_idx de c_orders. Ainsi, vous pouvez accéder à l'index 0 de c_orders.

```
CREATE MATERIALIZED VIEW super_mv distkey(c_custkey) sortkey(c_custkey) AS (
  SELECT c_custkey, o.o_orderstatus, o.o_totalprice, o_idx
  FROM customer_orders_lineitem c, c.c_orders o AT o_idx
);
```

Les attributs o_orderstatus et o_totalprice de la vue matérialisée super_mv sont SUPER.

La vue matérialisée super_mv sera actualisée de manière incrémentielle lors des modifications apportées à la table de base customer_orders_lineitem.

```
REFRESH MATERIALIZED VIEW super_mv;
INFO: Materialized view super_mv was incrementally updated successfully.
```

Pour réécrire la première requête PartiQL en tant que requête SQL standard, joignez customer_orders_lineitem avec super_mv comme suit.

```
SELECT c.c_name, v.o_orderstatus
FROM customer_orders_lineitem c
JOIN super_mv v ON c.c_custkey = v.c_custkey;
```

De la même manière, vous pouvez réécrire la deuxième requête PartiQL. L'exemple suivant utilise un filtre sur o_idx = 0.

```
SELECT c.c_name, v.o_totalprice
FROM customer_orders_lineitem c
JOIN super_mv v ON c.c_custkey = v.c_custkey
WHERE v.o_idx = 0;
```

Dans l'instruction CREATE MATERIALISED VIEW, spécifiez c_custkey comme clé de distribution et clé de tri pour super_mv. Amazon Redshift effectue une fusion efficace, en supposant que c_custkey est également la clé de distribution et la clé de tri de customer_orders_lineitem. Si ce n'est pas le cas, vous pouvez spécifier c_custkey comme clé de tri et clé de distribution de customer_orders_lineitem comme suit.

```
ALTER TABLE customer_orders_lineitem
ALTER DISTKEY c_custkey, ALTER SORTKEY (c_custkey);
```

Utilisez l'instruction EXPLAIN pour vérifier qu'Amazon Redshift effectue une fusion sur les requêtes réécrites.

```
EXPLAIN
      SELECT c.c_name, v.o_orderstatus
      FROM customer_orders_lineitem c JOIN super_mv v ON c.c_custkey = v.c_custkey;

      QUERY PLAN

 ------------------------------------------------------------------------------------------
      XN Merge Join DS_DIST_NONE  (cost=0.00..34701.82 rows=1470776 width=27)
      Merge Cond: ("outer".c_custkey = "inner".c_custkey)
      ->  XN Seq Scan on mv_tbl__super_mv__0 derived_table2  (cost=0.00..14999.86
 rows=1499986 width=13)
      ->  XN Seq Scan on customer_orders_lineitem c  (cost=0.00..999.96 rows=99996
 width=30)
      (4 rows)
```

Création de colonnes scalaires Amazon Redshift à partir de données fragmentées

Les données sans schéma stockées dans SUPER peuvent affecter les performances d'Amazon Redshift. Par exemple, les prédicats de filtrage ou les conditions de jointure en tant qu'analyse à plage restreinte ne peuvent pas utiliser efficacement les cartes de zone. Les utilisateurs et les outils BI peuvent utiliser des vues matérialisées comme présentation conventionnelle des données et augmenter les performances des requêtes analytiques.

La requête suivante analyse la vue matérialisée `super_mv` et les filtres sur `o_orderstatus`.

```
SELECT c.c_name, v.o_totalprice
FROM customer_orders_lineitem c
JOIN super_mv v ON c.c_custkey = v.c_custkey
WHERE v.o_orderstatus = 'F';
```

Inspectez `stl_scan` pour vérifier que Amazon Redshift ne peut effectivement pas utiliser les cartes de zone sur le scan à plage restreinte sur `o_orderstatus`.

```
SELECT slice, is_rrscan FROM stl_scan
WHERE query = pg_last_query_id() AND perm_table_name LIKE '%super_mv%';

 slice | is_rrscan
-------+-----------
     0 | f
     1 | f
     5 | f
     4 | f
     2 | f
     3 | f
(6 rows)
```

L'exemple suivant adapte la vue matérialisée `super_mv` pour créer des colonnes scalaires à partir des données fragmentées. Dans ce cas, Amazon Redshift convertit `o_orderstatus` de SUPER à VARCHAR. En outre, spécifiez `o_orderstatus` comme clé de tri pour `super_mv`.

```
CREATE MATERIALIZED VIEW super_mv distkey(c_custkey) sortkey(c_custkey, o_orderstatus)
 AS (
  SELECT c_custkey, o.o_orderstatus::VARCHAR AS o_orderstatus, o.o_totalprice, o_idx
  FROM customer_orders_lineitem c, c.c_orders o AT o_idx
);
```

Après avoir réexécuté la requête, vérifiez qu'Amazon Redshift peut désormais utiliser des cartes de zones.

```
SELECT v.o_totalprice
FROM super_mv v
WHERE v.o_orderstatus = 'F';
```

Vous pouvez vérifier que l'analyse à plage restreinte utilise désormais les cartes de zones comme suit.

```
SELECT slice, is_rrscan FROM stl_scan
WHERE query = pg_last_query_id() AND perm_table_name LIKE '%super_mv%';

 slice | is_rrscan
-------+-----------
     0 | t
     1 | t
     2 | t
     3 | t
     4 | t
     5 | t
(6 rows)
```

Limites de l'utilisation du type de données SUPER avec les vues matérialisées

Avec Amazon Redshift, vous pouvez créer des vues matérialisées avec le type de données SUPER afin de précalculer des requêtes complexes et d'améliorer les performances des requêtes. Le type de données SUPER vous permet de stocker le jeu de résultats d'une requête sous forme de structure de données plate, ce qui permet un accès et un traitement plus rapides.

Les sections suivantes fournissent des détails sur les limites et les meilleures pratiques relatives à l'utilisation du type de données SUPER avec des vues matérialisées dans Amazon Redshift.

Les vues matérialisées dans Amazon Redshift n'ont aucune restriction spécifique en ce qui concerne PartiQL ou SUPER.

Pour plus d'informations sur les limitations générales de SQL lors de la création de vues matérialisées, consultez Limites.

Pour plus d'informations sur les limitations générales de SQL concernant le rafraîchissement progressif des vues matérialisées, consultez Limites d'actualisation incrémentielle.

Machine learning

Le machine learning d'Amazon Redshift (Amazon Redshift ML) est un service robuste, basé sur le cloud, qui facilite l'utilisation de la technologie de machine learning par les analystes et les scientifiques des données de tous niveaux de compétence. Amazon Redshift ML utilise un modèle pour générer des résultats. Vous pouvez utiliser les modèles de différentes manières :

- Vous pouvez fournir les données que vous souhaitez pour entraîner un modèle, ainsi que les métadonnées associées aux entrées de données à Amazon Redshift. Amazon Redshift ML crée ensuite des modèles dans Amazon SageMaker AI qui capturent des modèles dans les données d'entrée. En utilisant vos propres données pour le modèle, vous pouvez utiliser Amazon Redshift ML pour identifier les tendances des données, telles que la prévision du taux de désabonnement, la valeur du cycle de vie du client ou la prévision des revenus. Vous pouvez utiliser ces modèles pour générer des prévisions pour les nouvelles données d'entrée sans encourir de coûts supplémentaires.
- Vous pouvez utiliser l'un des modèles de base (FM) fournis par Amazon Bedrock, tels que Claude ou Amazon Titan. À l'aide d'Amazon Bedrock, vous pouvez associer la puissance de grands modèles linguistiques (LLMs) à vos données d'analyse dans Amazon Redshift en quelques étapes. En utilisant un modèle de langage étendu (LLM) externe, vous pouvez utiliser Amazon Redshift pour effectuer le traitement automatique du langage naturel (NLP) sur vos données. Vous pouvez utiliser le NLP pour des applications telles que la génération de texte, l'analyse des sentiments ou la traduction. Pour plus d'informations sur l'utilisation d'Amazon Bedrock avec Amazon Intégration d'Amazon Redshift ML à Amazon Bedrock Redshift, consultez.

> **Note**
>
> Refus d'utiliser vos données pour améliorer le service
> Si vous utilisez des modèles Amazon Bedrock, nous vous encourageons à lire les AWS politiques relatives à la manière dont le service Amazon Bedrock gère vos données. Vous devez déterminer si vous devez utiliser une politique de désinscription pour empêcher le service d'utiliser vos données pour améliorer le modèle ou le service, si Amazon Bedrock implémente une telle fonctionnalité à l'avenir. Pour vous assurer que le service n'utilise pas vos données à de telles fins, utilisez la politique générale de AWS désinscription.
> Pour plus d’informations, consultez les ressources suivantes :
>
> - Politiques de désabonnement aux services d'IA

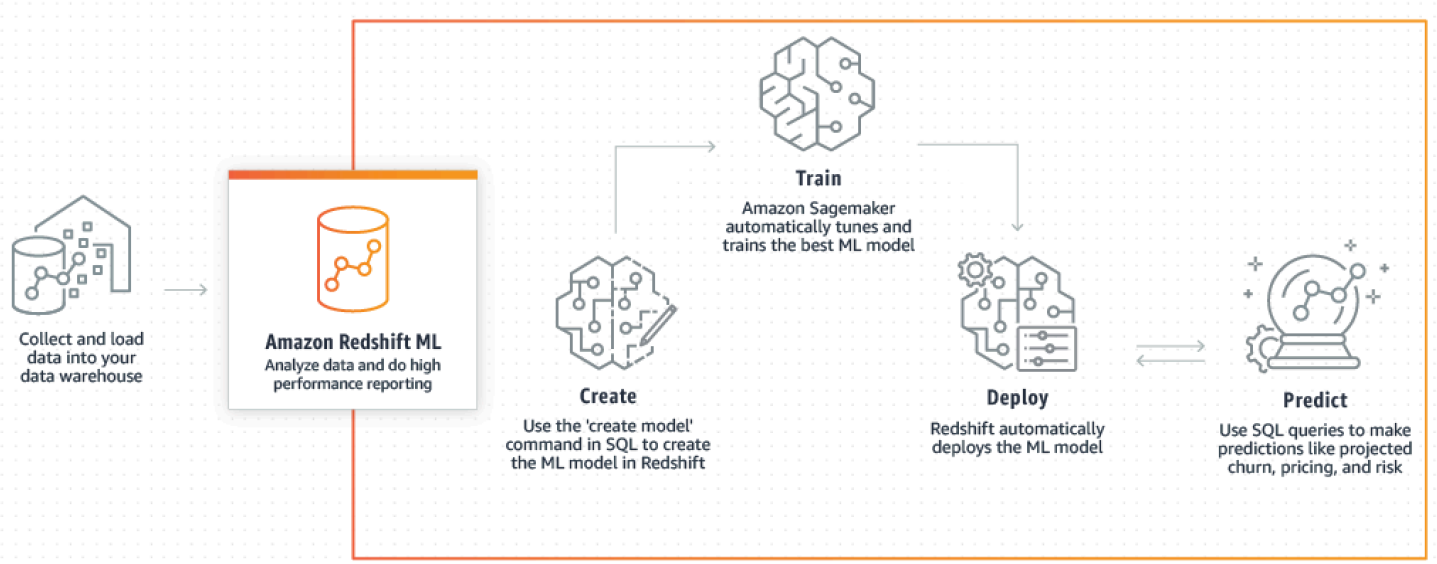
- Amazon Bedrock FAQs

> **Note**
>
> LLMs peut générer des informations inexactes ou incomplètes. Nous vous recommandons de vérifier les informations LLMs produites pour vous assurer qu'elles sont exactes et complètes.

Comment Amazon Redshift ML fonctionne avec Amazon AI SageMaker

Amazon Redshift fonctionne avec Amazon SageMaker AI Autopilot pour obtenir automatiquement le meilleur modèle et rendre la fonction de prédiction disponible dans Amazon Redshift.

Le schéma suivant illustre le fonctionnement d'Amazon Redshift ML.



De manière générale, procédez comme suit :

1. Amazon Redshift exporte les données d'entraînement vers Amazon S3.
2. Amazon SageMaker AI Autopilot prétraite les données d'entraînement. Le prétraitement remplit des fonctions importantes, telles que l'imputation des valeurs manquantes. Il reconnaît que certaines colonnes sont catégoriques (comme le code postal), les formate correctement pour l'entraînement, et effectue de nombreuses autres tâches. Choisir les meilleurs préprocesseurs à appliquer à l'ensemble de données d'entraînement est un problème en soi, et Amazon SageMaker AI Autopilot automatise sa solution.

3. Amazon SageMaker AI Autopilot trouve l'algorithme et les hyperparamètres de l'algorithme qui fournissent au modèle les prédictions les plus précises.
4. Amazon Redshift enregistre la fonction de prédiction en tant que fonction SQL dans votre cluster Amazon Redshift.
5. Lorsque vous exécutez des instructions CREATE MODEL, Amazon Redshift utilise Amazon SageMaker AI pour la formation. Par conséquent, il y a un coût associé pour l'entraînement de votre modèle. Il s'agit d'une rubrique distincte pour Amazon SageMaker AI dans votre AWS facture. Vous payez également les frais de stockage de vos données d'entraînement dans Amazon S3. L'inférence qui utilise des modèles créés avec CREATE MODEL pouvant être compilés et exécutés sur votre cluster Redshift ne sera pas facturée. L'utilisation d'Amazon Redshift ML n'entraîne pas de frais supplémentaires pour Amazon Redshift.

Rubriques

- Présentation du machine learning
- Le machine learning pour les novices et les experts
- Coûts d'utilisation d'Amazon Redshift ML
- Premiers pas avec Amazon Redshift ML
- Tutoriels pour Amazon Redshift ML
- Intégration d'Amazon Redshift ML à Amazon Bedrock

Présentation du machine learning

Avec Amazon Redshift, vous pouvez tirer parti des fonctionnalités d'apprentissage automatique pour obtenir des informations précieuses à partir de vos données. Cette présentation du Machine Learning (ML) vous montre comment explorer, visualiser et préparer vos données pour la formation et le déploiement de modèles ML. Les sections suivantes vous guideront tout au long du processus d'utilisation d'Amazon Redshift ML pour exploiter le potentiel de vos données grâce au machine learning.

En utilisant Amazon Redshift ML, vous pouvez entraîner des modèles de machine learning en utilisant des instructions SQL et les invoquer dans des requêtes SQL pour la prédiction.

Pour vous aider à apprendre à utiliser Amazon Redshift ML, vous pouvez regarder la vidéo suivante : Amazon Redshift ML.

Pour plus d'informations sur les conditions requises pour configurer votre cluster Redshift ou votre groupe de travail sans serveur, les autorisations et les droits de propriété pour utiliser Amazon Redshift ML, consultez les sections suivantes. Ces sections décrivent également le fonctionnement de l'entraînement et des prédictions simples dans Amazon Redshift ML.

Comment le machine learning peut résoudre un problème

Un modèle de machine learning génère des prédictions en trouvant des modèles dans vos données d'entraînement et en appliquant ensuite ces modèles à de nouvelles données. Dans le machine learning, vous entraînez ces modèles en apprenant les schémas qui expliquent le mieux vos données. Ensuite, vous utilisez les modèles pour élaborer des prédictions (également appelées inférences) sur de nouvelles données. Le machine learning est typiquement un processus itératif où vous pouvez continuer à améliorer la précision des prédictions en changeant les paramètres et en améliorant vos données de formation. Si les données changent, il faut réentraîner de nouveaux modèles avec le nouveau jeu de données.

Pour répondre à divers objectifs métier, il existe différentes approches fondamentales en matière de machine learning.

Apprentissage supervisé dans Amazon Redshift ML

Amazon Redshift prend en charge l'apprentissage supervisé, qui est l'approche la plus courante en matière d'analytique avancée d'entreprise. L'apprentissage supervisé est l'approche privilégiée du machine learning lorsque vous disposez d'un jeu de données établi et que vous comprenez comment des données d'entrée spécifiques prédisent divers résultats métier. Ces résultats sont parfois appelés étiquettes. En particulier, votre jeu de données est une table avec des attributs qui comprennent des caractéristiques (entrées) et des cibles (sorties). Par exemple, supposons que vous ayez une table qui fournit l'âge et le code postal des clients anciens et actuels. Supposons que vous ayez également un champ « active » (actif) qui est défini sur true pour les clients actuels et sur false pour les clients qui ont suspendu leur adhésion. L'objectif du machine learning supervisé est de repérer les modèles d'âge et de code postal menant à la perte de clients, tels que représentés par les clients dont les cibles ont la valeur « False ». Vous pouvez utiliser ce modèle pour prédire les clients susceptibles de se désabonner, par exemple en suspendant leur adhésion, et éventuellement offrir des incitatifs de rétention.

Amazon Redshift prend en charge l'apprentissage supervisé qui inclut la régression, la classification binaire et la classification multiclasse. La régression fait référence au problème de la prédiction de valeurs continues, telles que les dépenses totales des clients. La classification binaire fait référence

au problème de la prédiction de l'un des deux résultats, comme la prédiction de la résiliation ou non d'un client. La classification multiclasse fait référence au problème de la prédiction d'un résultat parmi plusieurs, comme la prédiction de l'article qui pourrait intéresser un client. Les analystes et les scientifiques des données peuvent l'utiliser pour effectuer un apprentissage supervisé afin de s'attaquer à des problèmes allant de la prédiction à la personnalisation, en passant par la prédiction du taux de désabonnement des clients. Vous pouvez également utiliser l'apprentissage supervisé pour les problèmes tels que la prédiction des ventes qui seront conclues, la prédiction des revenus, la détection de fraude et la prédiction de la valeur de la durée de vie des clients.

Apprentissage non supervisé dans Amazon Redshift ML

L'apprentissage non supervisé utilise des algorithmes de machine learning pour analyser et regrouper des données de formation non étiquetées. Les algorithmes découvrent des modèles ou des regroupements cachés. L'objectif est de modéliser la structure ou la distribution sous-jacente dans les données pour en savoir plus sur les données.

Amazon Redshift prend en charge l'algorithme de mise en cluster K-Means pour résoudre un problème d'apprentissage non supervisé. Cet algorithme résout les problèmes de mise en cluster lorsque vous souhaitez découvrir des regroupements dans les données. L'algorithme K-Means tente de trouver des regroupements discrets dans les données. Les données non classifiées sont regroupées et partitionnées en fonction de leurs similitudes et de leurs différences. Grâce au regroupement, l'algorithme K-Means détermine de manière itérative les meilleurs centroïdes et affecte chaque membre au centroïde le plus proche. Les membres les plus proches du même centroïde appartiennent au même groupe. Les membres d'un groupe sont aussi semblables que possible des autres membres du même groupe et aussi différents que possible des membres des autres groupes. Par exemple, l'algorithme de mise en cluster K-Means peut être utilisé pour classer les villes touchées par une pandémie ou en fonction de la popularité des produits de consommation.

Lorsque vous utilisez l'algorithme K-Means, vous spécifiez une entrée k qui indique le nombre de clusters à trouver dans les données. La sortie de cet algorithme est un ensemble de centroïdes k. Chaque point de données appartient à l'un des clusters k les plus proches. Chaque cluster est décrit par son centroïde. Le centroïde peut être considéré comme la moyenne multidimensionnelle du cluster. L'algorithme K-Means compare les distances pour voir à quel point les clusters sont différents les uns des autres. Une distance plus grande indique généralement une plus grande différence entre les clusters.

Le prétraitement des données est important pour K-Means, car il garantit que les fonctions du modèle restent à la même échelle et produisent des résultats fiables. Amazon Redshift prend en charge

certains préprocesseurs K-Means pour l'instruction CREATE MODEL, tels que StandardScaler, et. MinMax NumericPassthrough Si vous ne souhaitez appliquer aucun prétraitement pour K-means, choisissez-le NumericPassthrough explicitement en tant que transformateur. Pour plus d'informations sur les paramètres K-Means, consultez CREATE MODEL avec les paramètres K-MEANS.

Pour vous aider à apprendre à effectuer une formation non supervisée avec le clustering K-Means, vous pouvez regarder la vidéo suivante : Formation non supervisée avec le clustering K-Means.

Termes et concepts pour Amazon Redshift ML

Les termes suivants sont utilisés pour décrire certains concepts d'Amazon Redshift ML :

- Le machine learning dans Amazon Redshift entraîne un modèle avec une instruction SQL. Amazon Redshift ML et Amazon SageMaker AI gèrent toutes les conversions de données, les autorisations, l'utilisation des ressources et la découverte du modèle approprié.
- L'entraînement est la phase où Amazon Redshift crée un modèle de machine learning en exécutant un sous-jeu spécifié de données dans le modèle. Amazon Redshift lance automatiquement une tâche de formation dans Amazon SageMaker AI et génère un modèle.
- La prédiction (également appelée inférence) est l'utilisation du modèle dans les requêtes SQL Amazon Redshift pour prédire les résultats. Au moment de l'inférence, Amazon Redshift utilise une fonction de prédiction basée sur un modèle dans le cadre d'une requête plus vaste pour produire des prédictions. Les prédictions sont calculées localement, au niveau du cluster Redshift, offrant ainsi un débit élevé, une faible latence et aucuns frais.
- Avec Bring your own model (BYOM), vous pouvez utiliser un modèle formé en dehors d'Amazon Redshift avec Amazon AI pour effectuer des inférences dans la base de données localement dans SageMaker Amazon Redshift. Amazon Redshift ML prend en charge l'utilisation de BYOM dans l'inférence locale.
- L'inférence locale est utilisée lorsque les modèles sont préentraînés dans Amazon SageMaker AI, compilés par Amazon SageMaker AI Neo et localisés dans Amazon Redshift ML. Pour importer des modèles pris en charge pour l'inférence locale dans Amazon Redshift, utilisez l'instruction CREATE MODEL. Amazon Redshift importe les modèles d'IA préentraînés en appelant Amazon SageMaker SageMaker AI Neo. Vous y compilez le modèle et importez le modèle compilé dans Amazon Redshift. Utilisez l'inférence locale pour accélérer la vitesse et réduire les coûts.
- L'inférence à distance est utilisée lorsqu'Amazon Redshift invoque un modèle de point de terminaison déployé dans l'IA. SageMaker L'inférence à distance offre la flexibilité d'invoquer tous les types de modèles personnalisés et de modèles d'apprentissage profond, tels que les TensorFlow modèles que vous avez créés et déployés dans Amazon SageMaker AI.

Les éléments suivants sont également importants :

- Amazon SageMaker AI est un service d'apprentissage automatique entièrement géré. Avec Amazon SageMaker AI, les data scientists et les développeurs peuvent facilement créer, former et déployer directement des modèles dans un environnement hébergé prêt pour la production. Pour plus d'informations sur Amazon SageMaker AI, consultez la section Qu'est-ce qu'Amazon SageMaker AI dans le manuel Amazon SageMaker AI Developer Guide.
- Amazon SageMaker AI Autopilot est un ensemble de fonctionnalités qui entraîne et règle automatiquement les meilleurs modèles d'apprentissage automatique pour la classification ou la régression, en fonction de vos données. Vous conservez le plein contrôle et une visibilité totale. Amazon SageMaker AI Autopilot prend en charge les données d'entrée au format tabulaire. Amazon SageMaker AI Autopilot assure le nettoyage et le prétraitement automatiques des données, la sélection automatique d'algorithmes pour la régression linéaire, la classification binaire et la classification multiclasse. Il prend également en charge l'optimisation automatique des hyperparamètres (HPO), l'entraînement distribué, ainsi que les instances automatiques et la sélection de la taille des clusters. Pour plus d'informations sur Amazon SageMaker AI Autopilot, consultez Automatiser le développement de modèles avec Amazon SageMaker AI Autopilot dans le manuel Amazon SageMaker AI Developer Guide.
- Amazon Bedrock est un service entièrement géré qui propose un choix de modèles de base très performants (FMs) proposés par des sociétés d'IA de premier plan telles que AI21 Labs, Anthropic, Cohere, Meta, Mistral AI, Stability AI et Amazon via une seule API, ainsi qu'un large éventail de fonctionnalités nécessaires pour créer des applications d'IA génératives.

Le machine learning pour les novices et les experts

Avec Amazon Redshift, vous pouvez tirer parti des fonctionnalités du Machine Learning (ML) pour obtenir des informations à partir de vos données, que vous soyez novice ou expert en ML. Machine Learning est une fonctionnalité d'Amazon Redshift qui vous permet de créer, d'entraîner et de déployer des modèles de machine learning à l'aide de commandes SQL, sans avoir besoin d'une expertise approfondie en machine learning ou d'une ingénierie de données complexe.

Les sections suivantes vous guident tout au long du processus d'utilisation du Machine Learning, afin de vous permettre d'exploiter tout le potentiel de vos données avec Amazon Redshift.

Amazon Redshift ML vous permet d'entraîner des modèles avec une seule instruction SQL CREATE MODEL. L'instruction CREATE MODEL crée un modèle qu'Amazon Redshift utilise pour générer des prédictions basées sur un modèle avec des constructions SQL familières.

Amazon Redshift ML est particulièrement utile lorsque vous n'avez aucune expertise en matière d'apprentissage automatique, d'outils, de langages, d'algorithmes et. APIs Avec Amazon Redshift ML, vous n'avez pas à effectuer les opérations lourdes et indifférenciées nécessaires à l'intégration d'un service externe de machine learning. Amazon Redshift vous fait gagner du temps pour formater et déplacer les données, gérer les contrôles d'autorisation ou créer des intégrations, des flux de travail et des scripts personnalisés. Vous pouvez facilement utiliser les algorithmes de machine learning les plus populaires et simplifier les besoins en entraînement qui nécessitent une itération fréquente de l'entraînement à la prédiction. Amazon Redshift détermine automatiquement le meilleur algorithme et affine le meilleur modèle correspondant à votre problème. Vous pouvez faire des prédictions à partir du cluster Amazon Redshift sans avoir à déplacer les données en dehors d'Amazon Redshift ni à vous interfacer avec un autre service et à payer pour celui-ci.

Amazon Redshift ML soutient les analystes de données et les scientifiques des données dans l'utilisation du machine learning. Il permet également aux experts du machine learning d'utiliser leurs connaissances pour guider l'instruction CREATE MODEL afin d'utiliser uniquement les aspects qu'ils spécifient. En procédant ainsi, vous pouvez accélérer le temps dont CREATE MODEL a besoin pour trouver le meilleur candidat et/ou améliorer la précision du modèle.

L'instruction CREATE MODEL offre une certaine souplesse dans la façon dont vous pouvez spécifier les paramètres de la tâche d'entraînement. Cette flexibilité permet aux utilisateurs novices ou experts en machine learning de choisir leurs préprocesseurs, algorithmes, types de problèmes ou hyperparamètres préférés. Par exemple, un utilisateur intéressé par le taux de désabonnement peut spécifier dans l'instruction CREATE MODEL que le type de problème est une classification binaire qui fonctionne bien pour le taux de désabonnement. Ensuite, l'instruction CREATE MODEL réduit sa recherche du meilleur modèle à ceux de type classification binaire. Même si l'utilisateur choisit le type de problème, il existe encore de nombreuses options à utiliser avec l'instruction CREATE MODEL. Par exemple, la fonction CREATE MODEL détermine et applique les meilleures transformations de prétraitement et sélectionne les meilleurs réglages d'hyperparamètres.

Amazon Redshift ML facilite la formation en trouvant automatiquement le meilleur modèle à l'aide d'Amazon SageMaker AI Autopilot. Dans les coulisses, Amazon SageMaker AI Autopilot entraîne et règle automatiquement le meilleur modèle d'apprentissage automatique en fonction des données que vous avez fournies. Amazon SageMaker AI Neo compile ensuite le modèle d'entraînement et le rend disponible à des fins de prédiction dans votre cluster Redshift. Lorsque vous exécutez une requête d'inférence basée sur le machine learning à l'aide d'un modèle entraîné, la requête peut utiliser toutes les capacités de traitement massivement parallèle d'Amazon Redshift. En même temps, la requête peut utiliser une prédiction basée sur le machine learning.

- Si vous débutez avec le machine learning et que vous avez une connaissance générale des différents aspects de celui-ci, tels que les préprocesseurs, les algorithmes et les hyperparamètres, utilisez l'instruction CREATE MODEL uniquement pour les aspects que vous spécifiez. Vous pouvez alors réduire le temps dont CREATE MODEL a besoin pour trouver le meilleur candidat ou améliorer la précision du modèle. En outre, vous pouvez augmenter la valeur opérationnelle des prédictions en introduisant des connaissances supplémentaires du domaine telles que le type de problème ou l'objectif. Par exemple, dans un scénario de désabonnement client, si le résultat « le client n'est pas actif » est rare, l'objectif F1 est souvent préféré à l'objectif Précision. Étant donné que les modèles à haute précision peuvent prédire « le client est actif » tout le temps, il en résulte une haute précision, mais peu de valeur opérationnelle. Pour plus d'informations sur les objectifs F1, consultez Auto MLJob Objective dans le manuel Amazon SageMaker AI API Reference.

  Pour plus d'informations sur les options de base de l'instruction CREATE MODEL, consultez CREATE MODEL simple.
- Si vous êtes un pratiquant avancé du machine learning, vous pouvez spécifier le type de problème et les préprocesseurs pour certaines fonctions (mais pas toutes). Ensuite, le modèle CREATE suit vos suggestions sur les aspects spécifiés. Dans le même temps, CREATE MODEL détecte les meilleurs préprocesseurs pour les fonctions restantes et les meilleurs hyperparamètres. Pour plus d'informations sur la façon de limiter un ou plusieurs aspects du pipeline d'entraînement, consultez CREATE MODEL avec guide de l’utilisateur.
- Si vous êtes expert en machine learning, vous pouvez prendre le contrôle total de l'entraînement et du réglage des hyperparamètres. L'instruction CREATE MODEL ne cherche pas à déterminer les préprocesseurs, algorithmes et hyperparamètres optimaux, car c'est vous qui faites tous les choix. Pour plus d'informations sur l'utilisation de l'instruction CREATE MODEL avec AUTO OFF, consultez CRÉEZ des XGBoost modèles avec AUTO OFF.
- En tant qu'ingénieur de données, vous pouvez utiliser un XGBoost modèle préentraîné dans Amazon SageMaker AI et l'importer dans Amazon Redshift pour une inférence locale. Avec Bring your own model (BYOM), vous pouvez utiliser un modèle formé en dehors d'Amazon Redshift avec Amazon AI pour effectuer des inférences dans la base de données localement dans SageMaker Amazon Redshift. Amazon Redshift ML prend en charge l'utilisation de BYOM en inférence locale ou distante.

  Pour plus d'informations sur l'utilisation de l'instruction CREATE MODEL pour une inférence locale ou distante, consultez Modèle BYOM (Bring Your Own Model) : inférence locale.

Si vous utilisez Amazon Redshift ML, vous pouvez choisir l'une des options suivantes pour entraîner et déployer votre modèle.

- Types de problèmes, voir CREATE MODEL avec guide de l'utilisateur.
- Objectifs, voir CREATE MODEL avec guide de l'utilisateur ou CRÉEZ des XGBoost modèles avec AUTO OFF.
- Types de modèles, voirCRÉEZ des XGBoost modèles avec AUTO OFF.
- Préprocesseurs, voir CREATE MODEL avec guide de l'utilisateur.
- Hyperparamètres, voir CRÉEZ des XGBoost modèles avec AUTO OFF.
- BYOM (Bring Your Own Model), voir Modèle BYOM (Bring Your Own Model) : inférence locale.

Coûts d'utilisation d'Amazon Redshift ML

Avec Amazon Redshift, vous pouvez tirer parti des fonctionnalités d'apprentissage automatique pour obtenir des informations à partir de vos données sans avoir besoin d'une expertise approfondie en ingénierie des données ou en apprentissage automatique. Les sections suivantes décrivent les coûts associés à l'utilisation d'Amazon Redshift ML, afin de vous aider à planifier et à optimiser vos dépenses tout en tirant parti de cette puissante intégration du machine learning.

Coûts liés à l'utilisation d'Amazon Redshift ML avec l'IA SageMaker

Amazon Redshift ML for SageMaker AI utilise les ressources de votre cluster existantes à des fins de prévision afin que vous puissiez éviter des frais supplémentaires liés à Amazon Redshift. Il n'y a pas de frais supplémentaires Amazon Redshift pour la création ou l'utilisation d'un modèle. La prédiction se produit localement dans votre cluster Redshift. Vous n'avez donc pas à payer de frais supplémentaires à moins que vous n'ayez besoin de redimensionner votre cluster. Amazon Redshift ML utilise Amazon SageMaker AI pour entraîner votre modèle, ce qui entraîne un coût supplémentaire.

Il n'y a pas de frais supplémentaires pour les fonctions de prédiction qui s'exécutent au sein de votre cluster Amazon Redshift. L'instruction CREATE MODEL utilise Amazon SageMaker AI et entraîne un coût supplémentaire. Le coût augmente avec le nombre de cellules dans vos données d'entraînement. Le nombre de cellules est le produit du nombre d'enregistrements (dans la requête d'entraînement ou dans la table) multiplié par le nombre de colonnes. Par exemple, lorsqu'une requête SELECT de l'instruction CREATE MODEL crée 10 000 enregistrements et 5 colonnes, le nombre de cellules qu'elle crée est de 50 000.

Dans certains cas, les données d'entraînement produites par la requête SELECT de CREATE MODEL dépassent la limite MAX_CELLS que vous avez indiquée (ou la limite par défaut de 1 million si vous n'en avez pas spécifiée). Dans ces cas, CREATE MODEL choisit au hasard la limite MAX_CELLS (c'est-à-dire le nombre d'enregistrements « colonnes » du jeu de données d'entraînement). CREATE MODEL effectue ensuite un entraînement à l'aide de ces tuples choisis au hasard. L'échantillonnage aléatoire garantit que le jeu de données d'entraînement réduit n'aura aucun biais. Ainsi, en définissant la limite MAX_CELLS, vous pouvez contrôler vos coûts d'entraînement.

Lorsque vous utilisez l'instruction de commande CREATE MODEL, vous pouvez utiliser les options MAX_CELLS et MAX_RUNTIME pour contrôler les coûts, le temps et la précision potentielle du modèle.

MAX_RUNTIME indique la durée maximale que l'entraînement peut prendre dans l' SageMaker IA lorsque l'option AUTO ON ou OFF est utilisée. Les tâches d'entraînement se terminent souvent plus tôt que MAX_RUNTIME, en fonction de la taille du jeu de données. Après l'entraînement d'un modèle, Amazon Redshift effectue une tâche supplémentaire en arrière-plan pour compiler et installer vos modèles dans votre cluster. Ainsi, l'exécution de CREATE MODEL peut prendre plus de temps que MAX_RUNTIME. Cependant, MAX_RUNTIME limite la quantité de calcul et le temps utilisés par l' SageMaker IA pour entraîner votre modèle. Vous pouvez vérifier l'état de votre modèle à tout moment en utilisant la commande SHOW MODEL.

Lorsque vous exécutez CREATE MODEL avec AUTO ON, Amazon Redshift ML utilise le pilote automatique SageMaker AI pour explorer automatiquement et intelligemment différents modèles (ou candidats) afin de trouver le meilleur. MAX_RUNTIME limite la durée et le nombre de calculs effectués. Si la valeur MAX_RUNTIME est trop basse, il se peut qu'il n'y ait pas assez de temps pour explorer ne serait-ce qu'un seul candidat. Si l'erreur « Autopilot candidate has no models (Le candidat Autopilot n'a pas de modèles) » s'affiche, réexécutez l'instruction CREATE MODEL avec une valeur MAX_RUNTIME plus élevée. Pour plus d'informations sur ce paramètre, consultez MaxAutoMLJobRuntimeInSecondsle manuel Amazon SageMaker AI API Reference.

Lorsque vous exécutez CREATE MODEL avec AUTO OFF, MAX_RUNTIME correspond à une limite de durée d'exécution de la tâche d'entraînement dans SageMaker l'IA. Les tâches d'entraînement se terminent souvent plus tôt, en fonction de la taille du jeu de données et des autres paramètres utilisés, tels que num_rounds dans MODEL_TYPE XGBOOST.

Vous pouvez également contrôler les coûts ou réduire le temps d'entraînement en spécifiant une valeur MAX_CELLS plus petite lorsque vous exécutez CREATE MODEL. Une cellule est une entrée dans la base de données. Chaque ligne correspond à autant de cellules qu'il y a de colonnes, qui peuvent être de largeur fixe ou variable. MAX_CELLS limite le nombre de cellules, et donc le nombre

d'exemples d'entraînement utilisés pour entraîner votre modèle. Par défaut, MAX_CELLS est défini à 1 million de cellules. La réduction de MAX_CELLS réduit le nombre de lignes issues du résultat de la requête SELECT dans CREATE MODEL qu'Amazon Redshift exporte et envoie SageMaker à AI pour entraîner un modèle. La réduction de MAX_CELLS permet donc de réduire la taille du jeu de données utilisé pour entraîner les modèles, que ce soit avec AUTO ON ou AUTO OFF. Cette approche permet de réduire les coûts et le temps d'entraînement des modèles. Pour consulter les informations relatives à la formation et aux délais de facturation d'une tâche de formation spécifique, choisissez Training jobs in Amazon SageMaker AI.

L'augmentation de MAX_RUNTIME et de MAX_CELLS améliore souvent la qualité des modèles en permettant à l' SageMaker IA d'explorer davantage de candidats. Ainsi, l' SageMaker IA peut prendre plus de temps pour former chaque candidat et utiliser davantage de données pour former de meilleurs modèles. Si vous souhaitez une itération ou une exploration plus rapide de votre jeu de données, utilisez des valeurs MAX_RUNTIME et MAX_CELLS plus basses. Si vous souhaitez améliorer la précision des modèles, utilisez des valeurs MAX_RUNTIME et MAX_CELLS plus élevées.

Pour plus d'informations sur les coûts associés aux différents nombres de cellules et sur les détails de l'offre gratuite, consultez la tarification Amazon Redshift.

Coûts d'utilisation d'Amazon Redshift ML avec Amazon Bedrock

L'utilisation d'Amazon Redshift ML avec Amazon Bedrock entraîne des coûts supplémentaires. Pour plus d’informations, consultez Tarification d’Amazon Bedrock.

Premiers pas avec Amazon Redshift ML

Amazon Redshift ML permet aux utilisateurs de SQL de créer, d'entraîner et de déployer facilement des modèles de machine learning à l'aide d'instructions SQL familières. Avec Amazon Redshift ML, vous pouvez utiliser les données de votre cluster Redshift pour entraîner des modèles avec Amazon AI. SageMaker Ensuite, les modèles sont localisés et les prédictions peuvent être faites dans une base de données Amazon Redshift. Amazon Redshift ML prend actuellement en charge les algorithmes d'apprentissage automatique suivants : XGBoost (AUTO ON et OFF), perceptron multicouche (AUTO ON), K-Means (AUTO OFF) et Linear Learner.

Rubriques

- Cluster et configuration de la mise en place pour l'administration Amazon Redshift ML
- Utilisation de l'explicabilité du modèle avec Amazon Redshift ML

- Métriques de probabilité Amazon Redshift ML

Cluster et configuration de la mise en place pour l'administration Amazon Redshift ML

Avant de travailler avec Amazon Redshift ML, terminez la configuration du cluster et configurez les autorisations pour utiliser Amazon Redshift ML.

Configuration du cluster pour l'utilisation d'Amazon Redshift ML

Avant de travailler avec Amazon Redshift ML, vous devez remplir les conditions préalables suivantes.

En tant qu'administrateur Amazon Redshift, effectuez la configuration unique suivante pour utiliser les clusters provisionnés par Amazon Redshift. Pour utiliser Amazon Redshift ML avec Amazon Redshift Serverless, consultez Getting started with Amazon Redshift Serverless.

Pour effectuer une configuration de cluster unique pour Amazon Redshift ML

1. Créez un cluster Redshift à l'aide du AWS Management Console ou du AWS Command Line Interface ()AWS CLI. Assurez-vous d'associer la politique AWS Identity and Access Management (IAM) lors de la création du cluster. Pour plus d'informations sur les autorisations requises pour utiliser Amazon Redshift ML avec Amazon SageMaker AI, consultez Autorisations requises pour utiliser Amazon Redshift machine learning (ML)
2. Créez le rôle IAM requis pour utiliser le machine learning Amazon Redshift de l'une des manières suivantes :

   - Pour utiliser l' SageMaker IA avec Amazon Redshift ML, créez un rôle IAM avec des `AmazonS3FullAccess` politiques. `AmazonSageMakerFullAccess` Si vous envisagez également de créer des modèles de prévisions, associez également la politique `AmazonForecastFullAccess` à votre rôle.
   - Pour utiliser Amazon Bedrock avec Amazon Redshift ML, créez un rôle `AmazonS3FullAccess` IAM avec des politiques. `AmazonBedrockFullAccess`
   - Nous vous recommandons de créer un rôle IAM via la console Amazon Redshift avec la politique `AmazonRedshiftAllCommandsFullAccess` disposant des autorisations nécessaires pour exécuter des commandes SQL, telles que CREATE MODEL. Amazon Redshift utilise un mécanisme transparent basé sur une API pour créer des rôles IAM par programme en votre nom. Compte AWS Amazon Redshift associe automatiquement les politiques AWS gérées existantes au rôle IAM. Cette approche signifie que vous pouvez rester

dans la console Amazon Redshift et que vous n'avez pas besoin de passer à la console IAM pour créer des rôles. Pour plus d'informations, consultez Création d'un rôle IAM par défaut pour Amazon Redshift.

Lorsqu'un rôle IAM est créé comme rôle par défaut pour votre cluster, incluez `redshift` dans le nom de la ressource ou utilisez une balise spécifique à RedShift pour étiqueter ces ressources.

Si le routage Amazon VPC amélioré est activé sur votre cluster, vous pouvez utiliser un rôle IAM créé via la console Amazon Redshift. Ce rôle IAM a la politique `AmazonRedshiftAllCommandsFullAccess` attachée et ajoute les autorisations suivantes à la politique. Ces autorisations supplémentaires permettent à Amazon Redshift de créer et de supprimer une Elastic Network Interface (ENI) dans votre compte et de l'associer à des tâches de compilation exécutées sur Amazon ou EC2 Amazon ECS. Cela permet d'accéder aux objets de vos compartiments Amazon S3 uniquement à partir d'un Virtual Private Cloud (VPC) avec un accès Internet bloqué.

```
{
    "Effect": "Allow",
    "Action": [
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeVpcs",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DeleteNetworkInterfacePermission",
        "ec2:DeleteNetworkInterface",
        "ec2:CreateNetworkInterfacePermission",
        "ec2:CreateNetworkInterface",
        "ec2:ModifyNetworkInterfaceAttribute"
    ],
    "Resource": "*"
}
```

Pour utiliser les modèles de fondation Amazon Bedrock, ajoutez la section suivante :

```
// Required section if you use Bedrock models.
{
   "Effect": "Allow",
```

```
    "Action": "bedrock:InvokeModel",
    "Resource": [
        "arn:aws:bedrock:<region>::foundation-model/*"
    ]
}
```

- Si vous souhaitez créer un rôle IAM avec une politique plus restrictive, vous pouvez utiliser celle qui suit. Vous pouvez également modifier cette politique afin de répondre à vos besoins.

  Le compartiment Amazon S3 `redshift-downloads/redshift-ml/` est l'emplacement où sont stockées les exemples de données utilisés pour les autres étapes et exemples. Vous pouvez le supprimer si vous n'avez pas besoin de charger des données à partir d'Amazon S3. Ou alors, remplacez-le par d'autres compartiments Amazon S3 que vous utilisez pour charger des données dans Amazon Redshift.

  Les valeurs *`your-account-id`*, *`your-role`* et `amzn-s3-demo-bucket` sont celles que vous spécifiez dans le cadre de votre instruction CREATE MODEL.

  (Facultatif) Utilisez la section AWS KMS clés de l'exemple de politique si vous spécifiez une AWS KMS clé lors de l'utilisation d'Amazon Redshift ML. La valeur *`your-kms-key`* est la clé que vous utilisez dans le cadre de votre instruction CREATE MODEL.

  ```
  {
      "Version": "2012-10-17",
      "Statement": [
          {
              "Effect": "Allow",
              "Action": [
                  "cloudwatch:PutMetricData",
                  "ecr:BatchCheckLayerAvailability",
                  "ecr:BatchGetImage",
                  "ecr:GetAuthorizationToken",
                  "ecr:GetDownloadUrlForLayer",
                  "logs:CreateLogGroup",
                  "logs:CreateLogStream",
                  "logs:DescribeLogStreams",
                  "logs:PutLogEvents",
                  "sagemaker:*Job*",
                  "sagemaker:AddTags",
                  "sagemaker:CreateModel",
  ```

```
                "sagemaker:CreateEndpoint",
                "sagemaker:CreateEndpointConfig",
                "sagemaker:DeleteEndpoint",
                "sagemaker:DeleteEndpointConfig,
                "sagemaker:DeleteModel"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole",
                "s3:AbortMultipartUpload",
                "s3:GetObject",
                "s3:DeleteObject",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:iam::<your-account-id>:role/<your-role>",
                "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "arn:aws:s3:::redshift-downloads/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket,
                "arn:aws:s3:::redshift-downloads"
            ]
        }
        // Optional section needed if you use AWS KMS keys.
        ,{
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant",
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:Encrypt",
                "kms:GenerateDataKey*"
             ],
```

```
            "Resource": [
               "arn:aws:kms:<your-region>:<your-account-id>:key/<your-kms-key>"
            ]
         }
      ]
}
```

3. Pour permettre à Amazon Redshift et à SageMaker AI d'assumer le rôle d'interaction avec d'autres services, ajoutez la politique de confiance suivante au rôle IAM.

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

4. (Facultatif) Créez un compartiment Amazon S3 et une AWS KMS clé. Ils sont destinés à Amazon Redshift pour stocker les données de formation envoyées à Amazon SageMaker AI et recevoir le modèle entraîné depuis Amazon SageMaker AI.
5. (Facultatif) Créez différentes combinaisons de rôles IAM et de compartiments Amazon S3 pour contrôler l'accès à différents groupes d'utilisateurs.
6. (Facultatif) Lorsque vous activez le routage VPC pour votre cluster Redshift, créez ce qui suit :

   - Un point de terminaison Amazon S3 ou le VPC dans lequel se trouve votre cluster Redshift
   - Un point de terminaison SageMaker IA ou un point de terminaison Amazon Bedrock VPC

   Cela permet au trafic de s'exécuter via votre VPC entre des services pendant l'exécution de la commande CREATE MODEL. Pour plus d'informations sur le routage VPC, consultez Routage VPC amélioré dans Amazon Redshift.

Pour plus d'informations sur les autorisations requises pour spécifier un VPC privé pour votre tâche de réglage des hyperparamètres, consultez Autorisations requises pour utiliser Amazon Redshift ML avec Amazon AI. SageMaker

> **Note**
>
> Les appels d'inférence effectués vers des modèles d' SageMaker IA distants ne passent pas par votre VPC.

Pour plus d'informations sur la manière d'utiliser l'instruction CREATE MODEL pour commencer à créer des modèles pour différents cas d'utilisation, consultez CREATE MODEL.

Gestion des autorisations et de la propriété

Tout comme pour d'autres objets de base de données, tels que des tables ou des fonctions, Amazon Redshift lie la création et l'utilisation de modèles ML pour accéder aux mécanismes de contrôle. Il existe des autorisations distinctes pour la création d'un modèle exécutant des fonctions de prédiction.

Les exemples suivants illustrent comment Amazon Redshift gère le contrôle d'accès en utilisant deux groupes d'utilisateurs, `retention_analyst_grp` (créateur du modèle) et `marketing_analyst_grp` (utilisateur du modèle). L'analyste de rétention crée des modèles de machine learning que d'autres groupes d'utilisateurs peuvent utiliser grâce aux autorisations acquises.

Un super-utilisateur peut accorder l'autorisation GRANT USER ou GROUP pour créer des modèles de machine learning à l'aide de l'instruction suivante.

```
GRANT CREATE MODEL TO GROUP retention_analyst_grp;
```

Les utilisateurs ou groupes disposant de cette autorisation peuvent créer un modèle dans n'importe quel schéma du cluster si un utilisateur dispose de l'autorisation CREATE habituelle sur le SCHEMA. Le modèle de machine learning fait partie de la hiérarchie du schéma de la même manière que les tables, les vues, les procédures et les fonctions définies par l'utilisateur.

En supposant qu'un schéma `demo_ml` existe déjà, accordez l'autorisation aux deux groupes d'utilisateurs sur le schéma comme suit.

```
GRANT CREATE, USAGE ON SCHEMA demo_ml TO GROUP retention_analyst_grp;
```

```
GRANT USAGE ON SCHEMA demo_ml TO GROUP marketing_analyst_grp;
```

Vous pouvez permettre à d'autres utilisateurs d'utiliser votre fonction d'inférence du machine learning en leur accordant l'autorisation EXECUTE. L'exemple suivant utilise l'autorisation EXECUTE pour accorder au groupe marketing_analyst_grp l'autorisation d'utiliser le modèle.

```
GRANT EXECUTE ON MODEL demo_ml.customer_churn_auto_model TO GROUP
 marketing_analyst_grp;
```

Utilisez l'instruction REVOKE avec CREATE MODEL et EXECUTE pour révoquer ces autorisations auprès d'utilisateurs ou de groupes. Pour plus d'informations sur les instructions de contrôle d'autorisations, consultez GRANT et REVOKE.

Utilisation de l'explicabilité du modèle avec Amazon Redshift ML

Avec l'explicabilité du modèle dans Amazon Redshift ML, vous utilisez des valeurs d'importance des fonctions pour aider à comprendre comment chaque attribut de vos données d'entraînement contribue au résultat prédit.

L'explicabilité du modèle aide à améliorer vos modèles de machine learning (ML) en expliquant les prédictions réalisées par vos modèles. L'explicabilité du modèle vous explique comment les modèles réalisent des prédictions à l'aide d'une approche d'attribution de fonctions.

Amazon Redshift ML intègre l'explicabilité du modèle pour fournir une fonctionnalité d'explication du modèle aux utilisateurs Amazon Redshift ML. Pour plus d'informations sur l'explicabilité des modèles, consultez Qu'est-ce que l'équité et l'explicabilité des modèles pour les prédictions du Machine Learning ? dans le manuel Amazon SageMaker AI Developer Guide.

L'explicabilité du modèle surveille également les inférences produites par les modèles en production, pour la dérive dans l'attribution de fonctions. Elle fournit également les outils nécessaires pour générer des rapports de gouvernance de modèles destinés aux équipes responsables des risques et de la conformité, et aux organismes de réglementation externes.

Lorsque vous spécifiez l'option AUTO ON ou AUTO OFF lorsque vous utilisez l'instruction CREATE MODEL, une fois la tâche d'entraînement du modèle terminée, l' SageMaker IA crée la sortie d'explication. Vous pouvez utiliser la fonction EXPLAIN_MODEL pour interroger le rapport

d'explicabilité au format JSON. Pour de plus amples informations, veuillez consulter Solutions de machine learning.

Métriques de probabilité Amazon Redshift ML

Dans les problèmes d'apprentissage supervisé, les étiquettes de classe sont le résultat de prédictions utilisant les données d'entrée. Par exemple, si vous utilisez un modèle pour prédire si un client se réabonnera à un service de streaming, les étiquettes possibles sont probable et peu probable. Redshift ML offre la possibilité d'utiliser des métriques de probabilité, qui attribuent une probabilité à chaque étiquette pour indiquer sa probabilité. Cela vous permet de prendre des décisions plus éclairées en fonction des résultats prévus. Dans Amazon Redshift ML, des métriques de probabilité sont disponibles lors de la création de modèles AUTO ON présentant un type de problème de classification binaire ou de classification multi-classes. Si vous omettez le paramètre AUTO ON, Redshift ML suppose que le modèle doit disposer de la fonction AUTO ON.

Créer le modèle

Lors de la création d'un modèle, Amazon Redshift détecte automatiquement le type de modèle et le type de problème. S'il s'agit d'un problème de classification, Redshift crée automatiquement une deuxième fonction d'inférence que vous pouvez utiliser pour générer des probabilités relatives à chaque étiquette. Le nom de cette deuxième fonction d'inférence est le nom de la fonction d'inférence que vous avez spécifié, suivi de la chaîne `_probabilities`. Par exemple, si vous nommez votre fonction d'inférence `customer_churn_predict`, le nom de la deuxième fonction d'inférence est `customer_churn_predict_probabilities`. Vous pouvez ensuite interroger cette fonction pour obtenir les probabilités de chaque étiquette.

```
CREATE MODEL customer_churn_model
FROM customer_activity
    PROBLEM_TYPE BINARY_CLASSIFICATION
TARGET churn
FUNCTION customer_churn_predict
IAM_ROLE {default}
AUTO ON
SETTINGS ( S3_BUCKET 'amzn-s3-demo-bucket'
```

Obtenir des probabilités

Une fois que la fonction de probabilité est prête, l'exécution de la commande renvoie un type SUPER qui contient des tableaux des probabilités renvoyées et leurs étiquettes associées. Par exemple, le

résultat `"probabilities" : [0.7, 0.3], "labels" : ["False.", "True."]` signifie que l'étiquette False a une probabilité de 0,7 et que l'étiquette True a une probabilité de 0,3.

```
SELECT customer_churn_predict_probabilities(Account_length, Area_code,
            VMail_message, Day_mins, Day_calls, Day_charge,Eve_mins, Eve_calls,
            Eve_charge, Night_mins, Night_calls, Night_charge,Intl_mins, Intl_calls,
            Intl_charge, Cust_serv_calls)
FROM customer_activity;

customer_churn_predict_probabilities
 --------------------
 {"probabilities" : [0.7, 0.3], "labels" : ["False.", "True."]}
 {"probabilities" : [0.8, 0.2], "labels" : ["False.", "True."]}
 {"probabilities" : [0.75, 0.25], "labels" : ["True.", "False"]}
```

Les tableaux de probabilités et d'étiquettes sont toujours triés en fonction de leurs probabilités par ordre décroissant. Vous pouvez écrire une requête pour renvoyer uniquement l'étiquette prédite présentant la probabilité la plus élevée en désimbriquant les résultats renvoyés par SUPER par la fonction de probabilité.

```
SELECT prediction.labels[0], prediction.probabilities[0]
            FROM (SELECT customer_churn_predict_probabilities(Account_length,
 Area_code,
            VMail_message, Day_mins, Day_calls, Day_charge,Eve_mins, Eve_calls,
            Eve_charge, Night_mins, Night_calls, Night_charge,Intl_mins, Intl_calls,
            Intl_charge, Cust_serv_calls) AS prediction
FROM customer_activity);

  labels   | probabilities
-----------+--------------
 "False."  | 0.7
 "False."  | 0.8
 "True."   | 0.75
```

Pour simplifier les requêtes, vous pouvez stocker les résultats de la fonction de prédiction dans une table.

```
CREATE TABLE churn_auto_predict_probabilities AS
             (SELECT customer_churn_predict_probabilities(Account_length, Area_code,
             VMail_message, Day_mins, Day_calls, Day_charge,Eve_mins, Eve_calls,
             Eve_charge, Night_mins, Night_calls, Night_charge,Intl_mins,
```

```
             Intl_calls, Intl_charge, Cust_serv_calls) AS prediction
FROM customer_activity);
```

Vous pouvez interroger la table contenant les résultats pour renvoyer uniquement les prédictions dont la probabilité est supérieure à 0,7.

```
SELECT prediction.labels[0], prediction.probabilities[0]
FROM churn_auto_predict_probabilities
WHERE prediction.probabilities[0] > 0.7;

  labels   | probabilities
-----------+--------------
 "False."  | 0.8
 "True."   | 0.75
```

En utilisant la notation d'index, vous pouvez obtenir la probabilité d'une étiquette spécifique. L'exemple suivant renvoie les probabilités de toutes les étiquettes `True.`.

```
SELECT label, index, p.prediction.probabilities[index]
FROM churn_auto_predict_probabilities p, p.prediction.labels AS label AT index
WHERE label='True.';

  label  | index | probabilities
---------+-------+---------------
 "True." |     0 | 0.3
 "True." |     0 | 0.2
 "True." |     0 | 0.75
```

L'exemple suivant renvoie toutes les lignes portant une étiquette `True` avec une probabilité supérieure à 0,7, ce qui indique que le client est susceptible de se désister.

```
SELECT prediction.labels[0], prediction.probabilities[0]
FROM churn_auto_predict_probabilities
WHERE prediction.probabilities[0] > 0.7 AND prediction.labels[0] = "True.";

labels     | probabilities
-----------+--------------
 "True."   | 0.75
```

Tutoriels pour Amazon Redshift ML

Vous pouvez utiliser Amazon Redshift ML pour entraîner des modèles de machine learning en utilisant des instructions SQL et les invoquer dans des requêtes SQL pour la prédiction. Le machine learning dans Amazon Redshift entraîne un modèle avec une instruction SQL. Amazon Redshift lance automatiquement une tâche de formation dans Amazon SageMaker AI et génère un modèle. Une fois qu'un modèle est créé, vous pouvez effectuer des prédictions dans Amazon Redshift à l'aide de la fonction de prédiction du modèle.

Suivez les étapes décrites dans ces tutoriels pour en savoir plus sur les fonctionnalités Amazon Redshift ML :

- Tutoriel : Création de modèles de désabonnement des clients— Dans ce didacticiel, vous utilisez Amazon Redshift ML pour créer un modèle de désabonnement client à l'aide de la commande CREATE MODEL et exécuter des requêtes de prédiction pour des scénarios utilisateur. Ensuite, vous implémentez des requêtes à l'aide de la fonction SQL générée par la commande CREATE MODEL.
- Tutoriel : Création de modèles de clustering en k-moyennes— Dans ce didacticiel, vous utiliserez Amazon Redshift ML pour créer, entraîner et déployer un modèle d'apprentissage automatique basé sur l'algorithme K-means.
- Tutoriel : Création de modèles de classification multiclasse— Dans ce didacticiel, vous utiliserez Amazon Redshift ML pour créer un modèle d'apprentissage automatique qui résout les problèmes de classification multiclasses. L'algorithme de classification multiclasse classe les points de données entre trois classes ou plus. Ensuite, vous implémentez des requêtes à l'aide de la fonction SQL générée par la commande CREATE MODEL.
- Tutoriel : Construire XGBoost des modèles— Dans ce didacticiel, vous allez créer un modèle à partir des données d'Amazon S3 et exécuter des requêtes de prédiction avec le modèle à l'aide d'Amazon Redshift ML. L' XGBoost algorithme est une implémentation optimisée de l'algorithme des arbres amplifiés par le gradient.
- Tutoriel : Création de modèles de régression— Dans ce didacticiel, vous utiliserez Amazon Redshift ML pour créer un modèle de régression basé sur le machine learning et exécuter des requêtes de prédiction sur le modèle. Les modèles de régression vous permettent de prédire des résultats numériques, tels que le prix d'une maison ou le nombre de personnes qui utiliseront le service de location de vélos d'une ville.
- Tutoriel : Création de modèles de régression avec apprentissage linéaire— Dans ce didacticiel, vous allez créer un modèle d'apprenant linéaire à partir des données d'Amazon S3 et exécuter des

requêtes de prédiction avec le modèle à l'aide d'Amazon Redshift ML. L'algorithme d'apprentissage linéaire basé sur l' SageMaker IA résout les problèmes de régression ou de classification multi-classes.

- Tutoriel : Création de modèles de classification multiclasse avec apprentissage linéaire— Dans ce didacticiel, vous allez créer un modèle d'apprenant linéaire à partir des données d'Amazon S3, puis exécuter des requêtes de prédiction avec le modèle à l'aide d'Amazon Redshift ML. L'algorithme d'apprentissage linéaire basé sur l' SageMaker IA résout les problèmes de régression ou de classification.

Tutoriel : Création de modèles de désabonnement des clients

Dans ce tutoriel, vous utilisez Amazon Redshift ML pour créer un modèle de désabonnement des clients à l'aide de la commande CREATE MODEL et vous exécutez des requêtes de prédiction pour les scénarios utilisateur. Ensuite, vous implémentez des requêtes à l'aide de la fonction SQL générée par la commande CREATE MODEL.

Vous pouvez utiliser une simple instruction CREATE MODEL pour exporter des données d'entraînement, entraîner un modèle, importer le modèle et préparer une fonction de prédiction Amazon Redshift. Utilisez l'instruction CREATE MODEL pour spécifier les données d'entraînement sous forme de table ou d'instruction SELECT.

Cet exemple utilise des informations d'historique pour créer un modèle de machine learning du taux de désabonnement des clients d'un opérateur mobile. Tout d'abord, l' SageMaker IA entraîne votre modèle d'apprentissage automatique, puis teste votre modèle à l'aide des informations de profil d'un client arbitraire. Une fois le modèle validé, Amazon SageMaker AI déploie le modèle et la fonction de prédiction sur Amazon Redshift. Vous pouvez utiliser la fonction de prédiction pour prédire si un client va se désabonner ou non.

Exemples de cas d'utilisation

Vous pouvez résoudre d'autres problèmes de classification binaire à l'aide d'Amazon Redshift ML, par exemple pour prédire si une vente sera conclue ou non. Vous pouvez également prédire si une transaction financière est frauduleuse ou non.

Tâches

- Prérequis
- Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift

- Étape 2 : Créer le modèle de machine learning
- Étape 3 : Effectuer des prédictions avec le modèle

Prérequis

Pour effectuer ce tutoriel, vous devez avoir rempli les conditions suivantes :

- Vous devez configurer un cluster Amazon Redshift pour Amazon Redshift ML. Pour cela, utilisez la documentation Configuration du cluster et de la mise en place pour l'administration Amazon Redshift ML.
- Le cluster Amazon Redshift que vous utilisez pour créer le modèle et le compartiment Amazon S3 que vous utilisez pour fournir les données d'entraînement et stocker les artefacts de modèle doivent être dans la même région AWS .
- Pour télécharger les commandes SQL et l'exemple de jeu de données utilisés dans cette documentation, effectuez l'une des opérations suivantes :
  - Télécharger les instructions SQL, le fichier d'activité client et le fichier Abalone.
  - À l'aide AWS CLI de for Amazon S3, exécutez la commande suivante. Vous pouvez utiliser votre propre chemin d'accès.

```
aws s3 cp s3://redshift-downloads/redshift-ml/tutorial-scripts/redshift-ml-
tutorial.sql </target/path>
aws s3 cp s3://redshift-downloads/redshift-ml/customer_activity/
customer_activity.csv </target/path>
aws s3 cp s3://redshift-downloads/redshift-ml/abalone_xgb/abalone_xgb.csv </target/
path>
```

Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift

Utilisez l'éditeur de requête v2 Amazon Redshift pour modifier et exécuter des requêtes, et visualiser les résultats.

L'exécution des requêtes suivantes crée une table nommée `customer_activity` et ingère l'exemple de jeu de données d'Amazon S3.

```
DROP TABLE IF EXISTS customer_activity;

CREATE TABLE customer_activity (
```

```
state varchar(2),
account_length int,
area_code int,
phone varchar(8),
intl_plan varchar(3),
vMail_plan varchar(3),
vMail_message int,
day_mins float,
day_calls int,
day_charge float,
total_charge float,
eve_mins float,
eve_calls int,
eve_charge float,
night_mins float,
night_calls int,
night_charge float,
intl_mins float,
intl_calls int,
intl_charge float,
cust_serv_calls int,
churn varchar(6),
record_date date
);

COPY customer_activity
FROM 's3://redshift-downloads/redshift-ml/customer_activity/'
REGION 'us-east-1' IAM_ROLE default
FORMAT AS CSV IGNOREHEADER 1;
```

Étape 2 : créer le modèle de machine learning

Le taux de désabonnement est notre entrée cible dans ce modèle. Toutes les autres entrées du modèle sont des attributs qui aident à créer une fonction permettant de prédire le taux de désabonnement.

L'exemple suivant utilise l'opération CREATE MODEL pour fournir un modèle qui prédit si un client sera actif, à l'aide d'entrées telles que l'âge, le code postal, les dépenses et les dossiers du client. Dans l'exemple suivant, remplacez amzn-s3-demo-bucket par votre propre compartiment Amazon S3.

```
CREATE MODEL customer_churn_auto_model
FROM
```

```
    (
      SELECT state,
             account_length,
             area_code,
             total_charge/account_length AS average_daily_spend,
             cust_serv_calls/account_length AS average_daily_cases,
             churn
      FROM customer_activity
      WHERE  record_date < '2020-01-01'
     )
TARGET churn FUNCTION ml_fn_customer_churn_auto
IAM_ROLE default SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket'
);
```

La requête SELECT de l'exemple précédent crée les données d'entraînement. La clause TARGET spécifie quelle colonne correspond à l'étiquette de machine learning que l'opération CREATE MODEL utilise pour apprendre à effectuer des prédictions. La colonne cible « churn » (désabonnement) indique si le client a toujours un abonnement actif ou s'il a suspendu l'adhésion. Le champ S3_BUCKET correspond au nom du compartiment Amazon S3 que vous avez créé précédemment. Le compartiment Amazon S3 est utilisé pour partager des données d'entraînement et des artefacts entre Amazon Redshift et Amazon SageMaker AI. Les colonnes restantes correspondent aux entités qui sont utilisées pour la prédiction.

Pour un résumé de la syntaxe et des entités d'un cas d'utilisation élémentaire de la commande CREATE MODEL, consultez CREATE MODEL simple.

Ajouter des autorisations pour le chiffrement côté serveur (facultatif)

Amazon Redshift utilise par défaut Amazon SageMaker AI Autopilot pour la formation. En particulier, Amazon Redshift exporte en toute sécurité les données d'entraînement dans le compartiment Amazon S3 spécifié par le client. Si vous ne spécifiez pas de KMS_KEY_ID, les données sont chiffrées par défaut à l'aide du chiffrement côté serveur SSE-S3.

Lorsque vous chiffrez vos entrées à l'aide d'un chiffrement côté serveur avec une clé AWS KMS gérée (SSE-MMS), ajoutez les autorisations suivantes :

```
{
    "Effect": "Allow",
    "Action": [
    "kms:Encrypt"
    "kms:Decrypt"
```

```
    ]
}
```

Pour plus d'informations sur les rôles Amazon SageMaker AI, consultez la section relative aux rôles Amazon SageMaker AI dans le manuel Amazon SageMaker AI Developer Guide.

Vérifier l'état de l'entraînement du modèle (facultatif)

Vous pouvez utiliser la commande SHOW MODEL pour savoir quand votre modèle sera prêt.

Utilisez l'opération suivante pour vérifier l'état du modèle.

```
SHOW MODEL customer_churn_auto_model;
```

Voici un exemple de la sortie de l'opération précédente.

```
+-------------------------
+-----------------------------------------------------------------------------------------------
+
|           Key            |
                  Value
             |
+-------------------------
+-----------------------------------------------------------------------------------------------
+
|        Model Name        |
        customer_churn_auto_model
             |
|       Schema Name        |
                 public
             |
|          Owner           |
                 awsuser
             |
|      Creation Time       |
        Tue, 14.06.2022 17:15:52
             |
|       Model State        |
                TRAINING
             |
|                          |

             |
```

```
|      TRAINING DATA:      |

             |
|          Query           | SELECT STATE, ACCOUNT_LENGTH, AREA_CODE, TOTAL_CHARGE /
 ACCOUNT_LENGTH AS AVERAGE_DAILY_SPEND, CUST_SERV_CALLS / ACCOUNT_LENGTH AS
 AVERAGE_DAILY_CASES, CHURN |
|                          |
         FROM CUSTOMER_ACTIVITY
             |
|                          |
    WHERE RECORD_DATE < '2020-01-01'
             |
|      Target Column       |
                  CHURN
             |
|                          |

             |
|       PARAMETERS:        |

             |
|        Model Type        |
                   auto
             |
|       Problem Type       |

             |
|        Objective         |

             |
|     AutoML Job Name      |
     redshiftml-20220614171552640901
             |
|      Function Name       |
        ml_fn_customer_churn_auto
             |
|   Function Parameters    |                                                 state
 account_length area_code average_daily_spend average_daily_cases
                       |
| Function Parameter Types |
      varchar int4 int4 float8 int4
             |
```

```
|         IAM Role         |
          default-aws-iam-role
             |
|        S3 Bucket         |
              amzn-s3-demo-bucket
               |
|       Max Runtime        |
                   5400
              |
+--------------------------
+-----------------------------------------------------------------------------------------------
+
```

Lorsque l'entraînement du modèle est terminé, la variable `model_state` devient `Model is Ready`, et la fonction de prédiction devient disponible.

Étape 3 : effectuer des prédictions avec le modèle

Vous pouvez utiliser des instructions SQL pour afficher les prédictions effectuées par le modèle de prédiction. Dans cet exemple, la fonction de prédiction créée par l'opération CREATE MODEL est nommée `ml_fn_customer_churn_auto`. Les arguments en entrée de la fonction de prédiction correspondent aux types d'entités, tels que varchar pour `state` et integer pour `account_length`. La sortie de la fonction de prédiction est du même type que celle de la colonne TARGET de l'instruction CREATE MODEL.

1. Vous avez entraîné le modèle sur des données datant d'avant le 01/01/2020. Vous utilisez donc maintenant la fonction de prédiction sur le jeu de test. La requête suivante affiche les prédictions indiquant si les clients qui se sont inscrits après le 01/01/2020 vont se désabonner ou non.

   ```
   SELECT
       phone,
       ml_fn_customer_churn_auto(
           state,
           account_length,
           area_code,
           total_charge / account_length,
           cust_serv_calls / account_length
       ) AS active
   FROM
       customer_activity
   WHERE
   ```

```
    record_date > '2020-01-01';
```

2. L'exemple suivant utilise la même fonction de prédiction pour un autre cas d'utilisation. Dans ce cas, Amazon Redshift prédit la proportion de clients qui se désabonnent et de ceux qui ne se désabonnent pas parmi les clients de différents états avec une date d'enregistrement ultérieure à 01/01/2020.

```
WITH predicted AS (
    SELECT
        state,
        ml_fn_customer_churn_auto(
            state,
            account_length,
            area_code,
            total_charge / account_length,
            cust_serv_calls / account_length
        ) :: varchar(6) AS active
    FROM
        customer_activity
    WHERE
        record_date > '2020-01-01'
)
SELECT
    state,
    SUM(
        CASE
            WHEN active = 'True.' THEN 1
            ELSE 0
        END
    ) AS churners,
    SUM(
        CASE
            WHEN active = 'False.' THEN 1
            ELSE 0
        END
    ) AS nonchurners,
    COUNT(*) AS total_per_state
FROM
    predicted
GROUP BY
    state
ORDER BY
    state;
```

3. L'exemple suivant utilise la fonction de prédiction pour le cas d'utilisation de la prédiction du pourcentage de clients qui se désabonnent dans un état. Dans ce cas, Amazon Redshift prédit le pourcentage de désabonnement quand la date d'enregistrement est ultérieure au 01/01/2020.

```
WITH predicted AS (
    SELECT
        state,
        ml_fn_customer_churn_auto(
            state,
            account_length,
            area_code,
            total_charge / account_length,
            cust_serv_calls / account_length
        ) :: varchar(6) AS active
    FROM
        customer_activity
    WHERE
        record_date > '2020-01-01'
)
SELECT
    state,
    CAST((CAST((SUM(
        CASE
            WHEN active = 'True.' THEN 1
            ELSE 0
        END
    )) AS FLOAT) / CAST(COUNT(*) AS FLOAT)) AS DECIMAL (3, 2)) AS pct_churn,
    COUNT(*) AS total_customers_per_state
FROM
    predicted
GROUP BY
    state
ORDER BY
    3 DESC;
```

Rubriques en relation

Pour plus d'informations sur Amazon Redshift ML, consultez la documentation suivante :

- Coûts d'utilisation d'Amazon Redshift ML
- Commande CREATE MODEL

- Fonction EXPLAIN_MODEL

Pour plus d'informations sur le machine learning, consultez la documentation suivante :

- Présentation du machine learning
- Machine learning pour les novices et les experts
- En quoi consistent l'équité et l'explicabilité des modèles pour les prédictions de machine learning ?

Tutoriel : Création de modèles de clustering en k-moyennes

Dans ce tutoriel, vous utilisez Amazon Redshift ML pour créer, entraîner et déployer un modèle de machine learning basé sur l'algorithme des k-moyennes. Cet algorithme résout les problèmes de mise en cluster lorsque vous souhaitez découvrir des regroupements dans les données. Les k-moyennes permettent de regrouper les données qui n'ont pas encore été étiquetées. Pour en savoir plus sur le clustering K-means, consultez How K-means Clustering Works dans le manuel Amazon SageMaker AI Developer Guide.

Vous allez utiliser une opération CREATE MODEL pour créer un modèle de k-moyennes à partir d'un cluster Amazon Redshift. Vous pouvez utiliser une commande CREATE MODEL pour exporter des données d'entraînement, entraîner un modèle, importer le modèle et préparer une fonction de prédiction Amazon Redshift. Utilisez l'opération CREATE MODEL pour spécifier les données d'entraînement sous la forme d'une table ou d'une instruction SELECT.

Dans ce tutoriel, vous utilisez l'algorithme des k-moyennes sur le jeu de données GDELT (Global Database of Events, Language, and Tone), qui suit l'actualité mondiale, et les données sont stockées pour chaque seconde de chaque jour. L'algorithme des k-moyennes regroupera les événements qui ont un ton, des acteurs ou des lieux similaires. Les données sont stockées sous forme de fichiers multiples sur Amazon Simple Storage Service, dans deux dossiers différents. Le dossier d'historique qui couvre les années 1979 à 2013, et le dossier des mises à jour quotidiennes, qui couvre les années à partir de 2013. Dans le cadre de cet exemple, nous utilisons le format historique et nous intégrons les données de 1979.

Exemples de cas d'utilisation

Vous pouvez résoudre d'autres problèmes de clustering avec Amazon Redshift ML, tels que le regroupement des clients ayant des habitudes de visionnage similaires sur un service de streaming. Vous pouvez également utiliser Redshift ML pour prédire le nombre optimal de centres d'expédition pour un service de livraison.

Tâches

- Prérequis
- Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift
- Étape 2 : Créer le modèle de machine learning
- Étape 3 : Effectuer des prédictions avec le modèle

Prérequis

Pour effectuer ce tutoriel, vous devez suivre la procédure Configuration administrative pour Amazon Redshift ML.

Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift

1. Utilisez l'éditeur de requête v2 Amazon Redshift pour exécuter la requête suivante. La requête supprime la table `gdelt_data` dans le schéma public si elle existe et crée une table du même nom dans le schéma public.

```
DROP TABLE IF EXISTS gdelt_data CASCADE;

CREATE TABLE gdelt_data (
    GlobalEventId bigint,
    SqlDate bigint,
    MonthYear bigint,
    Year bigint,
    FractionDate double precision,
    Actor1Code varchar(256),
    Actor1Name varchar(256),
    Actor1CountryCode varchar(256),
    Actor1KnownGroupCode varchar(256),
    Actor1EthnicCode varchar(256),
    Actor1Religion1Code varchar(256),
    Actor1Religion2Code varchar(256),
    Actor1Type1Code varchar(256),
    Actor1Type2Code varchar(256),
    Actor1Type3Code varchar(256),
    Actor2Code varchar(256),
    Actor2Name varchar(256),
    Actor2CountryCode varchar(256),
    Actor2KnownGroupCode varchar(256),
    Actor2EthnicCode varchar(256),
```

```
    Actor2Religion1Code varchar(256),
    Actor2Religion2Code varchar(256),
    Actor2Type1Code varchar(256),
    Actor2Type2Code varchar(256),
    Actor2Type3Code varchar(256),
    IsRootEvent bigint,
    EventCode bigint,
    EventBaseCode bigint,
    EventRootCode bigint,
    QuadClass bigint,
    GoldsteinScale double precision,
    NumMentions bigint,
    NumSources bigint,
    NumArticles bigint,
    AvgTone double precision,
    Actor1Geo_Type bigint,
    Actor1Geo_FullName varchar(256),
    Actor1Geo_CountryCode varchar(256),
    Actor1Geo_ADM1Code varchar(256),
    Actor1Geo_Lat double precision,
    Actor1Geo_Long double precision,
    Actor1Geo_FeatureID bigint,
    Actor2Geo_Type bigint,
    Actor2Geo_FullName varchar(256),
    Actor2Geo_CountryCode varchar(256),
    Actor2Geo_ADM1Code varchar(256),
    Actor2Geo_Lat double precision,
    Actor2Geo_Long double precision,
    Actor2Geo_FeatureID bigint,
    ActionGeo_Type bigint,
    ActionGeo_FullName varchar(256),
    ActionGeo_CountryCode varchar(256),
    ActionGeo_ADM1Code varchar(256),
    ActionGeo_Lat double precision,
    ActionGeo_Long double precision,
    ActionGeo_FeatureID bigint,
    DATEADDED bigint
);
```

2. La requête suivante charge les données d'exemple dans la table `gdelt_data`.

```
COPY gdelt_data
FROM 's3://gdelt-open-data/events/1979.csv'
REGION 'us-east-1'
```

```
IAM_ROLE default
CSV
DELIMITER '\t';
```

Examiner les données d'entraînement (facultatif)

Pour voir sur quelles données votre modèle sera entraîné, utilisez la requête suivante.

```
SELECT
    AvgTone,
    EventCode,
    NumArticles,
    Actor1Geo_Lat,
    Actor1Geo_Long,
    Actor2Geo_Lat,
    Actor2Geo_Long
FROM
    gdelt_data LIMIT 100;
```

Étape 2 : Créer le modèle de machine learning

L'exemple suivant utilise la commande CREATE MODEL pour créer un modèle qui regroupe les données en sept clusters. La valeur K correspond au nombre de clusters utilisés pour diviser vos points de données. Le modèle classe vos points de données en clusters dans lesquels les points de données sont plus similaires les uns aux autres. En partitionnant les points de données en groupes, l'algorithme des k-moyennes détermine de manière itérative le meilleur centre de cluster. L'algorithme attribue ensuite chaque point de données au centre de cluster le plus proche. Les membres les plus proches du même centre de cluster appartiennent au même groupe. Les membres d'un groupe sont aussi semblables que possible des autres membres du même groupe et aussi différents que possible des membres des autres groupes. La valeur K est subjective et dépend des méthodes qui mesurent les similitudes entre les points de données. Vous pouvez modifier la valeur K pour mieux égaliser les tailles des clusters si ces derniers sont distribués de manière inégale.

Dans l'exemple suivant, remplacez amzn-s3-demo-bucket par votre propre compartiment Amazon S3.

```
CREATE MODEL news_data_clusters
FROM
    (
        SELECT
```

```
            AvgTone,
            EventCode,
            NumArticles,
            Actor1Geo_Lat,
            Actor1Geo_Long,
            Actor2Geo_Lat,
            Actor2Geo_Long
        FROM
            gdelt_data
    ) FUNCTION news_monitoring_cluster
    IAM_ROLE default
    AUTO OFF
    MODEL_TYPE KMEANS
    PREPROCESSORS 'none'
    HYPERPARAMETERS DEFAULT
    EXCEPT
    (K '7')
    SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket');
```

Vérifier l'état de l'entraînement du modèle (facultatif)

Vous pouvez utiliser la commande SHOW MODEL pour savoir quand votre modèle est prêt.

Pour vérifier l'état du modèle, utilisez l'opération SHOW MODEL suivante et recherchez si `Model State` a pour valeur `Ready`.

```
SHOW MODEL NEWS_DATA_CLUSTERS;
```

Lorsque le modèle est prêt, la sortie de l'opération précédente doit indiquer que `Model State` a pour valeur `Ready`. Voici un exemple de la sortie de l'opération SHOW MODEL.

```
+-------------------------
+-----------------------------------------------------------------------------------------------
+
|        Model Name        |
 news_data_clusters                                          |
+-------------------------
+-----------------------------------------------------------------------------------------------
+
|       Schema Name        |                                                  public
                                          |
|          Owner           |                                                 awsuser
                                          |
```

```
|      Creation Time       |                                        Fri, 17.06.2022
 16:32:19                                  |
|       Model State        |                                           READY
                                          |
|        train:msd         |                                         2973.822754
                                          |
|      train:progress      |                                          100.000000
                                          |
|     train:throughput     |                                        237114.875000
                                          |
|      Estimated Cost      |                                           0.004983
                                          |
|                          |
                                          |
|      TRAINING DATA:      |
                                          |
|          Query           | SELECT AVGTONE, EVENTCODE, NUMARTICLES, ACTOR1GEO_LAT,
 ACTOR1GEO_LONG, ACTOR2GEO_LAT, ACTOR2GEO_LONG |
|                          |                                             FROM GDELT_DATA
                                          |
|                          |
                                          |
|       PARAMETERS:        |
                                          |
|        Model Type        |                                              kmeans
                                          |
|    Training Job Name     |
 redshiftml-20220617163219978978-kmeans                                 |
|      Function Name       |
 news_monitoring_cluster                                      |
|   Function Parameters    |       avgtone eventcode numarticles actor1geo_lat
 actor1geo_long actor2geo_lat actor2geo_long        |
| Function Parameter Types |                              float8 int8 int8 float8 float8
 float8 float8                            |
|         IAM Role         |                                        default-aws-iam-
role                                        |
|        S3 Bucket         |                                           amzn-s3-demo-
bucket                                      |
|       Max Runtime        |                                               5400
                                          |
|                          |
                                          |
|     HYPERPARAMETERS:     |
                                          |
```

```
|       feature_dim        |                                                 7
                                                |
|            k             |                                                 7
                                                |
+-------------------------
+-------------------------------------------------------------------------------------------------
+
```

Étape 3 : Effectuer des prédictions avec le modèle

Identifier les clusters

Vous pouvez trouver des regroupements discrets identifiés dans les données par votre modèle, également appelés clusters. Un cluster est l'ensemble des points de données qui sont plus proches de son centre de cluster que de tout autre centre de cluster. Puisque la valeur K représente le nombre de clusters dans le modèle, elle représente également le nombre de centres de cluster. La requête suivante identifie les clusters en montrant le cluster associé à chaque `globaleventid`.

```
SELECT
    globaleventid,
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS cluster
FROM
    gdelt_data;
```

Vérifier la distribution des données

Vous pouvez vérifier la distribution des données entre les clusters pour voir si la valeur K que vous avez choisie donne lieu à une distribution assez uniforme des données. Utilisez la requête suivante pour déterminer si les données sont distribuées uniformément entre vos clusters.

```
SELECT
    events_cluster,
    COUNT(*) AS nbr_events
FROM
```

```
    (
        SELECT
            globaleventid,
            news_monitoring_cluster(
                AvgTone,
                EventCode,
                NumArticles,
                Actor1Geo_Lat,
                Actor1Geo_Long,
                Actor2Geo_Lat,
                Actor2Geo_Long
            ) AS events_cluster
        FROM
            gdelt_data
    )
GROUP BY
    1;
```

Notez que vous pouvez modifier la valeur K pour mieux égaliser les tailles des clusters si ces derniers sont distribués de manière inégale.

Déterminer les centres de cluster

Un point de données est plus proche de son centre de cluster que de tout autre centre de cluster. Ainsi, la recherche des centres de cluster vous aide à définir les clusters.

Exécutez la requête suivante pour déterminer les centres des clusters en fonction du nombre d'articles par code d'événement.

```
SELECT
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS events_cluster,
    eventcode,
    SUM(numArticles) AS numArticles
FROM
    gdelt_data
```

```
GROUP BY
    1,
    2;
```

Afficher des informations sur les points de données d'un cluster

Utilisez la requête suivante pour renvoyer les données des points affectés au cinquième cluster. Les articles sélectionnés doivent avoir deux acteurs.

```
SELECT
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS events_cluster,
    eventcode,
    actor1name,
    actor2name,
    SUM(numarticles) AS totalarticles
FROM
    gdelt_data
WHERE
    events_cluster = 5
    AND actor1name <> ' '
    AND actor2name <> ' '
GROUP BY
    1,
    2,
    3,
    4
ORDER BY
    5 desc;
```

Afficher des données sur les événements avec des acteurs du même code ethnique

La requête suivante comptabilise le nombre d'articles écrits sur des événements avec un ton positif. La requête exige également que les deux acteurs aient le même code ethnique et elle renvoie le cluster auquel chaque événement est affecté.

```
SELECT
    news_monitoring_cluster (
        AvgTone,
        EventCode,
        NumArticles,
        Actor1Geo_Lat,
        Actor1Geo_Long,
        Actor2Geo_Lat,
        Actor2Geo_Long
    ) AS events_cluster,
    SUM(numarticles) AS total_articles,
    eventcode AS event_code,
    Actor1EthnicCode AS ethnic_code
FROM
    gdelt_data
WHERE
    Actor1EthnicCode = Actor2EthnicCode
    AND Actor1EthnicCode <> ' '
    AND Actor2EthnicCode <> ' '
    AND AvgTone > 0
GROUP BY
    1,
    3,
    4
HAVING
    (total_articles) > 4
ORDER BY
    1,
    2 ASC;
```

Rubriques en relation

Pour plus d'informations sur Amazon Redshift ML, consultez la documentation suivante :

- Coûts d'utilisation d'Amazon Redshift ML
- Opération CREATE MODEL
- Fonction EXPLAIN_MODEL

Pour plus d'informations sur le machine learning, consultez la documentation suivante :

- Présentation du machine learning

- Machine learning pour les novices et les experts
- En quoi consistent l'équité et l'explicabilité des modèles pour les prédictions de machine learning ?

Tutoriel : Création de modèles de classification multiclasse

Dans ce tutoriel, vous utilisez Amazon Redshift ML pour créer un modèle de machine learning capable de résoudre des problèmes de classification multiclasse. L'algorithme de classification multiclasse classe les points de données entre trois classes ou plus. Ensuite, vous implémentez des requêtes à l'aide de la fonction SQL générée par la commande CREATE MODEL.

Vous pouvez utiliser une commande CREATE MODEL pour exporter des données d'entraînement, entraîner un modèle, importer le modèle et préparer une fonction de prédiction Amazon Redshift. Utilisez l'opération CREATE MODEL pour spécifier les données d'entraînement sous la forme d'une table ou d'une instruction SELECT.

Pour suivre ce tutoriel, vous utilisez le jeu de données public E-Commerce Sales Forecast, qui inclut les données de vente d'un détaillant britannique en ligne. Le modèle que vous générez ciblera les clients les plus actifs en vue de les admettre à un programme de fidélité spécial pour la clientèle. Avec la classification multiclasse, vous pouvez utiliser ce modèle pour prédire le nombre de mois pendant lesquels un client sera actif sur une période de 13 mois. La fonction de prédiction désigne les clients qui devraient être actifs pendant 7 mois ou plus afin de les admettre au programme.

Exemples de cas d'utilisation

Vous pouvez résoudre d'autres problèmes de classification multiclasse avec Amazon Redshift ML, tels que la prédiction du produit le plus vendu d'une gamme de produits. Vous pouvez également prédire les fruits qu'une image contient, en sélectionnant par exemple des pommes, des poires ou des oranges.

Tâches

- Prérequis
- Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift
- Étape 2 : Créer le modèle de machine learning
- Étape 3 : Effectuer des prédictions avec le modèle

Prérequis

Pour effectuer ce tutoriel, vous devez suivre la procédure Configuration administrative pour Amazon Redshift ML.

Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift

Utilisez l'éditeur de requête v2 Amazon Redshift pour exécuter les requêtes suivantes. Ces requêtes chargent les données d'exemple dans Amazon Redshift.

1. La requête suivante crée une table nommée `ecommerce_sales`.

```
CREATE TABLE IF NOT EXISTS ecommerce_sales (
    invoiceno VARCHAR(30),
    stockcode VARCHAR(30),
    description VARCHAR(60),
    quantity DOUBLE PRECISION,
    invoicedate VARCHAR(30),
    unitprice DOUBLE PRECISION,
    customerid BIGINT,
    country VARCHAR(25)
);
```

2. La requête suivante copie les données d'exemple à partir du jeu de données E-Commerce Sales Forecast dans la table `ecommerce_sales`.

```
COPY ecommerce_sales
FROM
    's3://redshift-ml-multiclass/ecommerce_data.txt'
IAM_ROLE default
DELIMITER '\t'
IGNOREHEADER 1
REGION 'us-east-1'
MAXERROR 100;
```

Diviser les données

Lorsque vous créez un modèle dans Amazon Redshift ML, l' SageMaker IA divise automatiquement vos données en ensembles d'entraînement et de test, afin que l' SageMaker IA puisse déterminer la précision du modèle. En divisant manuellement les données dans cette étape, vous serez en mesure de vérifier la précision du modèle en allouant un jeu de prédiction supplémentaire.

Utilisez l'instruction SQL suivante pour diviser les données en trois jeux à des fins d'entraînement, de validation et de prédiction.

```
--creates table with all data
CREATE TABLE ecommerce_sales_data AS (
    SELECT
        t1.stockcode,
        t1.description,
        t1.invoicedate,
        t1.customerid,
        t1.country,
        t1.sales_amt,
        CAST(RANDOM() * 100 AS INT) AS data_group_id
    FROM
        (
            SELECT
                stockcode,
                description,
                invoicedate,
                customerid,
                country,
                SUM(quantity * unitprice) AS sales_amt
            FROM
                ecommerce_sales
            GROUP BY
                1,
                2,
                3,
                4,
                5
        ) t1
);

--creates training set
CREATE TABLE ecommerce_sales_training AS (
    SELECT
        a.customerid,
        a.country,
        a.stockcode,
        a.description,
        a.invoicedate,
        a.sales_amt,
        (b.nbr_months_active) AS nbr_months_active
```

```
    FROM
        ecommerce_sales_data a
        INNER JOIN (
            SELECT
                customerid,
                COUNT(
                    DISTINCT(
                        DATE_PART(y, CAST(invoicedate AS DATE)) || '-' || LPAD(
                            DATE_PART(mon, CAST(invoicedate AS DATE)),
                            2,
                            '00'
                        )
                    )
                ) AS nbr_months_active
            FROM
                ecommerce_sales_data
            GROUP BY
                1
        ) b ON a.customerid = b.customerid
    WHERE
        a.data_group_id < 80
);

--creates validation set
CREATE TABLE ecommerce_sales_validation AS (
    SELECT
        a.customerid,
        a.country,
        a.stockcode,
        a.description,
        a.invoicedate,
        a.sales_amt,
        (b.nbr_months_active) AS nbr_months_active
    FROM
        ecommerce_sales_data a
        INNER JOIN (
            SELECT
                customerid,
                COUNT(
                    DISTINCT(
                        DATE_PART(y, CAST(invoicedate AS DATE)) || '-' || LPAD(
                            DATE_PART(mon, CAST(invoicedate AS DATE)),
                            2,
                            '00'
```

```
                        )
                    )
                ) AS nbr_months_active
            FROM
                ecommerce_sales_data
            GROUP BY
                1
        ) b ON a.customerid = b.customerid
    WHERE
        a.data_group_id BETWEEN 80
        AND 90
);

--creates prediction set
CREATE TABLE ecommerce_sales_prediction AS (
    SELECT
        customerid,
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    FROM
        ecommerce_sales_data
    WHERE
        data_group_id > 90);
```

Étape 2 : Créer le modèle de machine learning

Dans cette étape, vous utilisez l'instruction CREATE MODEL pour créer votre modèle de machine learning à l'aide de la classification multiclasse.

La requête suivante crée le modèle de classification multiclasse avec le jeu d'entraînement à l'aide de l'opération CREATE MODEL. Remplacez amzn-s3-demo-bucket par votre propre compartiment Amazon S3.

```
CREATE MODEL ecommerce_customer_activity
FROM
    (
        SELECT
            customerid,
            country,
            stockcode,
```

```
            description,
            invoicedate,
            sales_amt,
            nbr_months_active
        FROM
            ecommerce_sales_training
    ) TARGET nbr_months_active FUNCTION predict_customer_activity IAM_ROLE default
 PROBLEM_TYPE MULTICLASS_CLASSIFICATION SETTINGS (
        S3_BUCKET 'amzn-s3-demo-bucket',
        S3_GARBAGE_COLLECT OFF
    );
```

Dans cette requête, vous spécifiez le type de problème `Multiclass_Classification`. La cible que vous prédisez pour le modèle est `nbr_months_active`. Lorsque l' SageMaker IA a fini d'entraîner le modèle, elle crée la fonction`predict_customer_activity`, que vous utiliserez pour faire des prédictions dans Amazon Redshift.

Afficher l'état de l'entraînement du modèle (facultatif)

Vous pouvez utiliser la commande SHOW MODEL pour savoir quand votre modèle sera prêt.

Utilisez la requête suivante pour renvoyer diverses métriques du modèle, y compris l'état et la précision du modèle.

```
SHOW MODEL ecommerce_customer_activity;
```

Lorsque le modèle est prêt, la sortie de l'opération précédente doit indiquer que `Model State` a pour valeur `Ready`. Voici un exemple de la sortie de l'opération SHOW MODEL.

```
+--------------------------
+-----------------------------------------------------------------------------------------
+
|        Model Name        |
 ecommerce_customer_activity                                |
+--------------------------
+-----------------------------------------------------------------------------------------
+
|       Schema Name        |                                                 public
                                  |
|          Owner           |                                                 awsuser
                                  |
```

```
|      Creation Time       |                                  Fri, 17.06.2022 19:02:15
                                    |
|       Model State        |                                            READY
                                    |
|   Training Job Status    |
 MaxAutoMLJobRuntimeReached                                |
|   validation:accuracy    |                                         0.991280
                                    |
|      Estimated Cost      |                                         7.897689
                                    |
|                          |
                                    |
|      TRAINING DATA:      |
                                    |
|          Query           | SELECT CUSTOMERID, COUNTRY, STOCKCODE, DESCRIPTION,
 INVOICEDATE, SALES_AMT, NBR_MONTHS_ACTIVE |
|                          |                                    FROM
 ECOMMERCE_SALES_TRAINING                                  |
|      Target Column       |                                        NBR_MONTHS_ACTIVE
                                    |
|                          |
                                    |
|       PARAMETERS:        |
                                    |
|        Model Type        |                                             xgboost
                                    |
|       Problem Type       |                                  MulticlassClassification
                                    |
|        Objective         |                                         Accuracy
                                    |
|     AutoML Job Name      |
 redshiftml-20220617190215268770                             |
|      Function Name       |
 predict_customer_activity                                 |
|   Function Parameters    |                customerid country stockcode description
 invoicedate sales_amt                |
| Function Parameter Types |                           int8 varchar varchar varchar
 varchar float8                          |
|         IAM Role         |                                    default-aws-iam-role
                                    |
|        S3 Bucket         |                                        amzn-s3-demo-
bucket                                     |
|       Max Runtime        |                                           5400
                                    |
```

```
+-------------------------
+-------------------------------------------------------------------------------------------------------
+
```

Étape 3 : Effectuer des prédictions avec le modèle

La requête suivante montre quels clients peuvent bénéficier de votre programme de fidélité client. Si le modèle prédit que le client sera actif pendant au moins 7 mois, il sélectionne le client pour le programme de fidélité.

```
SELECT
    customerid,
    predict_customer_activity(
        customerid,
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    ) AS predicted_months_active
FROM
    ecommerce_sales_prediction
WHERE
    predicted_months_active >= 7
GROUP BY
    1,
    2
LIMIT
    10;
```

Exécuter des requêtes de prédiction sur les données de validation (facultatif)

Exécutez les requêtes de prédiction suivantes par rapport aux données de validation pour voir le niveau de précision du modèle.

```
SELECT
    CAST(SUM(t1.match) AS decimal(7, 2)) AS predicted_matches,
    CAST(SUM(t1.nonmatch) AS decimal(7, 2)) AS predicted_non_matches,
    CAST(SUM(t1.match + t1.nonmatch) AS decimal(7, 2)) AS total_predictions,
    predicted_matches / total_predictions AS pct_accuracy
FROM
    (
```

```
        SELECT
            customerid,
            country,
            stockcode,
            description,
            invoicedate,
            sales_amt,
            nbr_months_active,
            predict_customer_activity(
                customerid,
                country,
                stockcode,
                description,
                invoicedate,
                sales_amt
            ) AS predicted_months_active,
            CASE
                WHEN nbr_months_active = predicted_months_active THEN 1
                ELSE 0
            END AS match,
            CASE
                WHEN nbr_months_active <> predicted_months_active THEN 1
                ELSE 0
            END AS nonmatch
        FROM
            ecommerce_sales_validation
    )t1;
```

Prédire le nombre de clients qui manquent une entrée (facultatif)

La requête suivante compare le nombre de clients qui ne devraient être actifs que 5 ou 6 mois. Le modèle prédit que ces clients ne bénéficieront pas du programme de fidélité. La requête compare ensuite le nombre de clients qui manquent de peu le programme au nombre de clients prédits comme admissibles au programme de fidélité. Cette requête pourrait être utilisée pour prendre une décision éclairée quant au fait d'abaisser ou non le seuil d'admission au programme de fidélité. Vous pouvez également déterminer s'il y a un nombre important de clients qui devraient manquer de peu l'admission au programme. Vous pourriez alors encourager ces clients à augmenter leur activité pour devenir membres du programme de fidélité.

```
SELECT
    predict_customer_activity(
        customerid,
```

```
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    ) AS predicted_months_active,
    COUNT(customerid)
FROM
    ecommerce_sales_prediction
WHERE
    predicted_months_active BETWEEN 5 AND 6
GROUP BY
    1
ORDER BY
    1 ASC
LIMIT
    10)
UNION
(SELECT
      NULL AS predicted_months_active,
    COUNT (customerid)
FROM
    ecommerce_sales_prediction
WHERE
    predict_customer_activity(
        customerid,
        country,
        stockcode,
        description,
        invoicedate,
        sales_amt
    ) >=7);
```

Rubriques en relation

Pour plus d'informations sur Amazon Redshift ML, consultez la documentation suivante :

- Coûts d'utilisation d'Amazon Redshift ML
- Opération CREATE MODEL
- Fonction EXPLAIN_MODEL

Pour plus d'informations sur le machine learning, consultez la documentation suivante :

- Présentation du machine learning
- Machine learning pour les novices et les experts
- En quoi consistent l'équité et l'explicabilité des modèles pour les prédictions de machine learning ?

Tutoriel : Construire XGBoost des modèles

Dans ce tutoriel, vous créez un modèle avec des données d'Amazon S3 et vous exécutez des requêtes de prédiction avec ce modèle à l'aide d'Amazon Redshift ML. L' XGBoost algorithme est une implémentation optimisée de l'algorithme des arbres amplifiés par le gradient. XGBoost gère plus de types de données, de relations et de distributions que les autres algorithmes d'arbres augmentés par gradient. Vous pouvez l'utiliser XGBoost pour les problèmes de régression, de classification binaire, de classification multiclasse et de classement. Pour plus d'informations sur l' XGBoostalgorithme, consultez l'XGBoostalgorithme dans le manuel Amazon SageMaker AI Developer Guide.

L'`CREATE MODEL`opération Amazon Redshift ML avec `AUTO OFF` cette option est actuellement prise en charge en XGBoost tant que. `MODEL_TYPE` Vous pouvez fournir des informations pertinentes telles que l'objectif et les hyperparamètres dans le cadre de la commande `CREATE MODEL`, en fonction de votre cas d'utilisation.

Dans ce tutoriel, vous utilisez le jeu de données banknote authentication Data Set, qui est un problème de classification binaire visant à prédire si un billet de banque donné est authentique ou contrefait.

Exemples de cas d'utilisation

Vous pouvez résoudre d'autres problèmes de classification binaire à l'aide d'Amazon Redshift ML, comme par exemple prédire si un patient est en bonne santé ou malade. Vous pouvez également prédire si un e-mail est un courrier indésirable ou non.

Tâches

- Prérequis
- Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift
- Étape 2 : Créer le modèle de machine learning
- Étape 3 : Effectuer des prédictions avec le modèle

Prérequis

Pour effectuer ce tutoriel, vous devez suivre la procédure Configuration administrative pour Amazon Redshift ML.

Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift

Utilisez l'éditeur de requête v2 Amazon Redshift pour exécuter les requêtes suivantes.

La requête suivante crée deux tables, charge les données depuis Amazon S3 et divise les données en un jeu d'entraînement et un jeu de test. Vous allez utiliser le jeu d'entraînement pour entraîner votre modèle et créer la fonction de prédiction. Ensuite, vous testerez la fonction de prédiction sur le jeu de test.

```
--create training set table
CREATE TABLE banknoteauthentication_train(
    variance FLOAT,
    skewness FLOAT,
    curtosis FLOAT,
    entropy FLOAT,
    class INT
);

--Load into training table
COPY banknoteauthentication_train
FROM
    's3://redshiftbucket-ml-sagemaker/banknote_authentication/train_data/' IAM_ROLE
 default REGION 'us-west-2' IGNOREHEADER 1 CSV;

--create testing set table
CREATE TABLE banknoteauthentication_test(
    variance FLOAT,
    skewness FLOAT,
    curtosis FLOAT,
    entropy FLOAT,
    class INT
);

--Load data into testing table
COPY banknoteauthentication_test
FROM
    's3://redshiftbucket-ml-sagemaker/banknote_authentication/test_data/'
    IAM_ROLE default
```

```
    REGION 'us-west-2'
    IGNOREHEADER 1
    CSV;
```

Étape 2 : Créer le modèle de machine learning

La requête suivante crée le XGBoost modèle dans Amazon Redshift ML à partir du kit d'apprentissage que vous avez créé à l'étape précédente. Remplacez `amzn-s3-demo-bucket` par votre propre `S3_BUCKET`, qui stockera vos jeux de données en entrée et d'autres artefacts Redshift ML.

```
CREATE MODEL model_banknoteauthentication_xgboost_binary
FROM
    banknoteauthentication_train
    TARGET class
    FUNCTION func_model_banknoteauthentication_xgboost_binary
    IAM_ROLE default
    AUTO OFF
    MODEL_TYPE xgboost
    OBJECTIVE 'binary:logistic'
    PREPROCESSORS 'none'
    HYPERPARAMETERS DEFAULT
EXCEPT(NUM_ROUND '100')
SETTINGS(S3_BUCKET 'amzn-s3-demo-bucket');
```

Afficher l'état de l'entraînement du modèle (facultatif)

Vous pouvez utiliser la commande SHOW MODEL pour savoir quand votre modèle sera prêt.

Utilisez la requête suivante pour surveiller la progression de l'entraînement du modèle.

```
SHOW MODEL model_banknoteauthentication_xgboost_binary;
```

Si le modèle est `READY`, l'opération SHOW MODEL fournit également la métrique `train:error`, comme illustré dans l'exemple suivant de la sortie. La métrique `train:error` est une mesure de la précision de votre modèle qui présente jusqu'à six décimales. Une valeur de 0 est la plus précise et une valeur de 1 est la moins précise.

```
+-------------------------+-------------------------------------------------+
|        Model Name       |   model_banknoteauthentication_xgboost_binary   |
```

```
+--------------------------+-----------------------------------------------------+
| Schema Name              | public                                              |
| Owner                    | awsuser                                             |
| Creation Time            | Tue, 21.06.2022 19:07:35                            |
| Model State              | READY                                               |
| train:error              |                                            0.000000 |
| Estimated Cost           |                                            0.006197 |
|                          |                                                     |
| TRAINING DATA:           |                                                     |
| Query                    | SELECT *                                            |
|                          | FROM "BANKNOTEAUTHENTICATION_TRAIN"                 |
| Target Column            | CLASS                                               |
|                          |                                                     |
| PARAMETERS:              |                                                     |
| Model Type               | xgboost                                             |
| Training Job Name        | redshiftml-20220621190735686935-xgboost             |
| Function Name            | func_model_banknoteauthentication_xgboost_binary    |
| Function Parameters      | variance skewness curtosis entropy                  |
| Function Parameter Types | float8 float8 float8 float8                         |
| IAM Role                 | default-aws-iam-role                                |
| S3 Bucket                | amzn-s3-demo-bucket                                 |
| Max Runtime              |                                                5400 |
|                          |                                                     |
| HYPERPARAMETERS:         |                                                     |
| num_round                |                                                 100 |
| objective                | binary:logistic                                     |
+--------------------------+-----------------------------------------------------+
```

Étape 3 : Effectuer des prédictions avec le modèle

Vérifier la précision du modèle

La requête de prédiction suivante utilise la fonction de prédiction créée à l'étape précédente pour vérifier la précision de votre modèle. Exécutez cette requête sur le jeu de test pour vous assurer que le modèle ne correspond pas trop au jeu d'entraînement. Cette correspondance étroite porte également le nom de surajustement, et un surajustement peut amener le modèle à effectuer des prédictions non fiables.

```
WITH predict_data AS (
    SELECT
        class AS label,
```

```
        func_model_banknoteauthentication_xgboost_binary (variance, skewness, curtosis,
 entropy) AS predicted,
        CASE
            WHEN label IS NULL THEN 0
            ELSE label
        END AS actual,
        CASE
            WHEN actual = predicted THEN 1 :: INT
            ELSE 0 :: INT
        END AS correct
    FROM
        banknoteauthentication_test
),
aggr_data AS (
    SELECT
        SUM(correct) AS num_correct,
        COUNT(*) AS total
    FROM
        predict_data
)
SELECT
    (num_correct :: FLOAT / total :: FLOAT) AS accuracy
FROM
    aggr_data;
```

Prédire la quantité de billets de banque authentiques et contrefaits

La requête de prédiction suivante renvoie la quantité prédite de billets authentiques et contrefaits dans le jeu de test.

```
WITH predict_data AS (
    SELECT
        func_model_banknoteauthentication_xgboost_binary(variance, skewness, curtosis,
 entropy) AS predicted
    FROM
        banknoteauthentication_test
)
SELECT
    CASE
        WHEN predicted = '0' THEN 'Original banknote'
        WHEN predicted = '1' THEN 'Counterfeit banknote'
        ELSE 'NA'
    END AS banknote_authentication,
```

```
    COUNT(1) AS count
FROM
    predict_data
GROUP BY
    1;
```

Trouver l'observation moyenne pour un billet de banque authentique et un faux billet

La requête de prédiction suivante renvoie la valeur moyenne de chaque entité pour les billets qui sont prédits comme authentiques ou contrefaits dans le jeu de test.

```
WITH predict_data AS (
    SELECT
        func_model_banknoteauthentication_xgboost_binary(variance, skewness, curtosis,
 entropy) AS predicted,
          variance,
          skewness,
          curtosis,
          entropy
    FROM
        banknoteauthentication_test
)
SELECT
    CASE
        WHEN predicted = '0' THEN 'Original banknote'
        WHEN predicted = '1' THEN 'Counterfeit banknote'
        ELSE 'NA'
    END AS banknote_authentication,
    TRUNC(AVG(variance), 2) AS avg_variance,
    TRUNC(AVG(skewness), 2) AS avg_skewness,
    TRUNC(AVG(curtosis), 2) AS avg_curtosis,
    TRUNC(AVG(entropy), 2) AS avg_entropy
FROM
    predict_data
GROUP BY
    1
ORDER BY
    2;
```

Rubriques en relation

Pour plus d'informations sur Amazon Redshift ML, consultez la documentation suivante :

- Coûts d'utilisation d'Amazon Redshift ML
- Opération CREATE MODEL
- Fonction EXPLAIN_MODEL

Pour plus d'informations sur le machine learning, consultez la documentation suivante :

- Présentation du machine learning
- Machine learning pour les novices et les experts
- En quoi consistent l'équité et l'explicabilité des modèles pour les prédictions de machine learning ?

Tutoriel : Création de modèles de régression

Dans ce tutoriel, vous utilisez Amazon Redshift ML pour créer un modèle de régression de machine learning et exécuter des requêtes de prédiction sur le modèle. Les modèles de régression vous permettent de prédire des résultats numériques, tels que le prix d'une maison ou le nombre de personnes qui utiliseront le service de location de vélos d'une ville. Vous utilisez la commande CREATE MODEL dans Amazon Redshift avec vos données d'entraînement. Ensuite, Amazon Redshift ML compile le modèle, importe le modèle entraîné dans Redshift et prépare une fonction de prédiction SQL. Vous pouvez utiliser la fonction de prédiction dans les requêtes SQL dans Amazon Redshift.

Dans ce tutoriel, vous allez utiliser Amazon Redshift ML pour créer un modèle de régression qui prédira le nombre de personnes qui utilisent le service de vélos en libre-service de la ville de Toronto à une heure donnée de la journée. Les entrées du modèle incluent les jours fériés et les conditions météorologiques. Vous allez utiliser un modèle de régression, car vous souhaitez obtenir un résultat numérique pour ce problème.

Vous pouvez utiliser la commande CREATE MODEL pour exporter des données d'entraînement, entraîner un modèle et mettre ce modèle à disposition dans Amazon Redshift en tant que fonction SQL. Utilisez l'opération CREATE MODEL pour spécifier les données d'entraînement sous la forme d'une table ou d'une instruction SELECT.

Exemples de cas d'utilisation

Vous pouvez résoudre d'autres problèmes de régression avec Amazon Redshift ML, tels que la prédiction de la valeur à vie d'un client. Vous pouvez également utiliser Redshift ML pour prédire le prix le plus rentable et le chiffre d'affaires qui en résulte pour un produit.

Tâches

- Prérequis
- Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift
- Étape 2 : Créer le modèle de machine learning
- Étape 3 : valider le modèle

Prérequis

Pour effectuer ce tutoriel, vous devez suivre la procédure Configuration administrative pour Amazon Redshift ML.

Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift

Utilisez l'éditeur de requête v2 Amazon Redshift pour exécuter les requêtes suivantes.

1. Vous devez créer trois tables pour charger les trois jeux de données publics dans Amazon Redshift. Les jeux de données sont Bike Share Toronto Ridership Data, Données climatiques historiques et Données historiques des jours fériés. Exécutez la requête suivante dans l'éditeur de requête Amazon Redshift pour créer des tables nommées `ridership`, `weather` et `holiday`.

```
CREATE TABLE IF NOT EXISTS ridership (
    trip_id INT,
    trip_duration_seconds INT,
    trip_start_time timestamp,
    trip_stop_time timestamp,
    from_station_name VARCHAR(50),
    to_station_name VARCHAR(50),
    from_station_id SMALLINT,
    to_station_id SMALLINT,
    user_type VARCHAR(20)
);

CREATE TABLE IF NOT EXISTS weather (
    longitude_x DECIMAL(5, 2),
    latitude_y DECIMAL(5, 2),
    station_name VARCHAR(20),
    climate_id BIGINT,
    datetime_utc TIMESTAMP,
    weather_year SMALLINT,
```

```
    weather_month SMALLINT,
    weather_day SMALLINT,
    time_utc VARCHAR(5),
    temp_c DECIMAL(5, 2),
    temp_flag VARCHAR(1),
    dew_point_temp_c DECIMAL(5, 2),
    dew_point_temp_flag VARCHAR(1),
    rel_hum SMALLINT,
    rel_hum_flag VARCHAR(1),
    precip_amount_mm DECIMAL(5, 2),
    precip_amount_flag VARCHAR(1),
    wind_dir_10s_deg VARCHAR(10),
    wind_dir_flag VARCHAR(1),
    wind_spd_kmh VARCHAR(10),
    wind_spd_flag VARCHAR(1),
    visibility_km VARCHAR(10),
    visibility_flag VARCHAR(1),
    stn_press_kpa DECIMAL(5, 2),
    stn_press_flag VARCHAR(1),
    hmdx SMALLINT,
    hmdx_flag VARCHAR(1),
    wind_chill VARCHAR(10),
    wind_chill_flag VARCHAR(1),
    weather VARCHAR(10)
);

CREATE TABLE IF NOT EXISTS holiday (holiday_date DATE, description VARCHAR(100));
```

2. La requête suivante charge les données d'exemple dans les tables que vous avez créées à l'étape précédente.

```
COPY ridership
FROM
    's3://redshift-ml-bikesharing-data/bike-sharing-data/ridership/'
    IAM_ROLE default
    FORMAT CSV
    IGNOREHEADER 1
    DATEFORMAT 'auto'
    TIMEFORMAT 'auto'
    REGION 'us-west-2'
    gzip;

COPY weather
```

```
FROM
    's3://redshift-ml-bikesharing-data/bike-sharing-data/weather/'
    IAM_ROLE default
    FORMAT csv
    IGNOREHEADER 1
    DATEFORMAT 'auto'
    TIMEFORMAT 'auto'
    REGION 'us-west-2'
    gzip;

COPY holiday
FROM
    's3://redshift-ml-bikesharing-data/bike-sharing-data/holiday/'
    IAM_ROLE default
    FORMAT csv
    IGNOREHEADER 1
    DATEFORMAT 'auto'
    TIMEFORMAT 'auto'
    REGION 'us-west-2'
    gzip;
```

3. La requête suivante effectue des transformations sur les jeux de données `ridership` et `weather` pour éliminer les biais ou les anomalies. La suppression des biais et des anomalies améliore la précision du modèle. La requête simplifie les tables en créant deux nouvelles vues appelées `ridership_view` et `weather_view`.

```
CREATE
OR REPLACE VIEW ridership_view AS
SELECT
    trip_time,
    trip_count,
    TO_CHAR(trip_time, 'hh24') :: INT trip_hour,
    TO_CHAR(trip_time, 'dd') :: INT trip_day,
    TO_CHAR(trip_time, 'mm') :: INT trip_month,
    TO_CHAR(trip_time, 'yy') :: INT trip_year,
    TO_CHAR(trip_time, 'q') :: INT trip_quarter,
    TO_CHAR(trip_time, 'w') :: INT trip_month_week,
    TO_CHAR(trip_time, 'd') :: INT trip_week_day
FROM
    (
        SELECT
            CASE
```

```
                WHEN TRUNC(r.trip_start_time) < '2017-07-01' :: DATE THEN
 CONVERT_TIMEZONE(
                    'US/Eastern',
                    DATE_TRUNC('hour', r.trip_start_time)
                )
                ELSE DATE_TRUNC('hour', r.trip_start_time)
            END trip_time,
            COUNT(1) trip_count
        FROM
            ridership r
        WHERE
            r.trip_duration_seconds BETWEEN 60
            AND 60 * 60 * 24
        GROUP BY
            1
    );

CREATE
OR REPLACE VIEW weather_view AS
SELECT
    CONVERT_TIMEZONE(
        'US/Eastern',
        DATE_TRUNC('hour', datetime_utc)
    ) daytime,
    ROUND(AVG(temp_c)) temp_c,
    ROUND(AVG(precip_amount_mm)) precip_amount_mm
FROM
    weather
GROUP BY
    1;
```

4. La requête suivante crée une table qui combine tous les attributs d'entrée pertinents de `ridership_view` et `weather_view` dans la table `trip_data`.

```
CREATE TABLE trip_data AS
SELECT
    r.trip_time,
    r.trip_count,
    r.trip_hour,
    r.trip_day,
    r.trip_month,
    r.trip_year,
    r.trip_quarter,
```

```
    r.trip_month_week,
    r.trip_week_day,
    w.temp_c,
    w.precip_amount_mm,CASE
        WHEN h.holiday_date IS NOT NULL THEN 1
        WHEN TO_CHAR(r.trip_time, 'D') :: INT IN (1, 7) THEN 1
        ELSE 0
    END is_holiday,
    ROW_NUMBER() OVER (
        ORDER BY
            RANDOM()
    ) serial_number
FROM
    ridership_view r
    JOIN weather_view w ON (r.trip_time = w.daytime)
    LEFT OUTER JOIN holiday h ON (TRUNC(r.trip_time) = h.holiday_date);
```

Afficher les données d'exemple (facultatif)

La requête suivante montre les entrées de la table. Vous pouvez exécuter cette opération pour vous assurer que la table a été créée correctement.

```
SELECT *
FROM trip_data
LIMIT 5;
```

Voici un exemple de la sortie de l'opération précédente.

```
+---------------------+------------+-----------+----------+------------+-----------
+-------------+-----------------+--------------+--------+-----------------
+------------+---------------+
|      trip_time      | trip_count | trip_hour | trip_day | trip_month | trip_year
 | trip_quarter | trip_month_week | trip_week_day | temp_c | precip_amount_mm |
 is_holiday | serial_number |
+---------------------+------------+-----------+----------+------------+-----------
+-------------+-----------------+--------------+--------+-----------------
+------------+---------------+
| 2017-03-21 22:00:00 |         47 |        22 |       21 |          3 |        17 |
         1 |               3 |             3 |      1 |                0 |          0 |
             1 |
```

```
| 2018-05-04 01:00:00 |         19 |         1 |        4 |           5 |        18 |
         2 |               1 |            6 |     12 |                0 |          0 |
             3 |
| 2018-01-11 10:00:00 |         93 |        10 |       11 |           1 |        18 |
         1 |               2 |            5 |      9 |                0 |          0 |
             5 |
| 2017-10-28 04:00:00 |         20 |         4 |       28 |          10 |        17 |
         4 |               4 |            7 |     11 |                0 |          1 |
             7 |
| 2017-12-31 21:00:00 |         11 |        21 |       31 |          12 |        17 |
         4 |               5 |            1 |    -15 |                0 |          1 |
             9 |
+---------------------+------------+-----------+----------+------------+-----------
+--------------+-----------------+---------------+--------+-----------------
+------------+--------------+
```

Montrer la corrélation entre les attributs (facultatif)

La détermination de la corrélation vous aide à mesurer la force de l'association entre les attributs. Le niveau d'association peut vous aider à déterminer ce qui affecte votre résultat cible. Dans ce tutoriel, le résultat cible est `trip_count`.

La requête suivante crée ou remplace la procédure `sp_correlation`. Vous utilisez la procédure stockée appelée `sp_correlation` pour montrer la corrélation entre un attribut et d'autres attributs d'une table dans Amazon Redshift.

```
CREATE OR REPLACE PROCEDURE sp_correlation(source_schema_name in varchar(255),
 source_table_name in varchar(255), target_column_name in varchar(255),
 output_temp_table_name inout varchar(255)) AS $$
DECLARE
  v_sql varchar(max);
  v_generated_sql varchar(max);
  v_source_schema_name varchar(255)=lower(source_schema_name);
  v_source_table_name varchar(255)=lower(source_table_name);
  v_target_column_name varchar(255)=lower(target_column_name);
BEGIN
  EXECUTE 'DROP TABLE IF EXISTS ' || output_temp_table_name;
  v_sql = '
SELECT
  ''CREATE temp table '|| output_temp_table_name||' AS SELECT ''|| outer_calculation||
  '' FROM (SELECT COUNT(1) number_of_items, SUM('||v_target_column_name||')
 sum_target, SUM(POW('||v_target_column_name||',2)) sum_square_target, POW(SUM('||
v_target_column_name||'),2) square_sum_target,''||
```

```
  inner_calculation||
  '' FROM (SELECT ''||
  column_name||
  '' FROM '||v_source_table_name||'))''
FROM
  (
  SELECT
    DISTINCT
    LISTAGG(outer_calculation,'','') OVER () outer_calculation
    ,LISTAGG(inner_calculation,'','') OVER () inner_calculation
    ,LISTAGG(column_name,'','') OVER () column_name
  FROM
    (
    SELECT
      CASE WHEN atttypid=16 THEN ''DECODE(''||column_name||'',true,1,0)'' ELSE
 column_name END column_name
      ,atttypid
      ,''CAST(DECODE(number_of_items * sum_square_''||rn||'' - square_sum_''||
rn||'',0,null,(number_of_items*sum_target_''||rn||'' - sum_target * sum_''||rn||
        '')/SQRT((number_of_items * sum_square_target - square_sum_target) *
 (number_of_items * sum_square_''||rn||
        '' - square_sum_''||rn||''))) AS numeric(5,2)) ''||column_name
 outer_calculation
      ,''sum(''||column_name||'') sum_''||rn||'',''||
            ''SUM(trip_count*''||column_name||'') sum_target_''||rn||'',''||
            ''SUM(POW(''||column_name||'',2)) sum_square_''||rn||'',''||
            ''POW(SUM(''||column_name||''),2) square_sum_''||rn inner_calculation
    FROM
      (
      SELECT
        row_number() OVER (order by a.attnum) rn
        ,a.attname::VARCHAR column_name
        ,a.atttypid
      FROM pg_namespace AS n
        INNER JOIN pg_class AS c ON n.oid = c.relnamespace
        INNER JOIN pg_attribute AS a ON c.oid = a.attrelid
      WHERE a.attnum > 0
        AND n.nspname = ''''||v_source_schema_name||'''
        AND c.relname = ''''||v_source_table_name||'''
        AND a.atttypid IN (16,20,21,23,700,701,1700)
      )
    )
)';
  EXECUTE v_sql INTO v_generated_sql;
```

```
  EXECUTE  v_generated_sql;
END;
$$ LANGUAGE plpgsql;
```

La requête suivante montre la corrélation entre la colonne cible, `trip_count`, et d'autres attributs numériques de notre jeu de données.

```
call sp_correlation(
    'public',
    'trip_data',
    'trip_count',
    'tmp_corr_table'
);

SELECT
    *
FROM
    tmp_corr_table;
```

Voici un exemple de la sortie de l'opération `sp_correlation` précédente.

```
+------------+-----------+----------+------------+-----------+--------------
+-----------------+---------------+--------+------------------+------------
+---------------+
| trip_count | trip_hour | trip_day | trip_month | trip_year | trip_quarter
 | trip_month_week | trip_week_day | temp_c | precip_amount_mm | is_holiday |
 serial_number |
+------------+-----------+----------+------------+-----------+--------------
+-----------------+---------------+--------+------------------+------------
+---------------+
|          1 |      0.32 |     0.01 |       0.18 |      0.12 |         0.18 |
     0 |          0.02 |   0.53 |            -0.07 |      -0.13 |             0 |
+------------+-----------+----------+------------+-----------+--------------
+-----------------+---------------+--------+------------------+------------
+---------------+
```

Étape 2 : Créer le modèle de machine learning

1. La requête suivante divise vos données en un jeu d'entraînement et un jeu de validation en désignant 80 % du jeu de données pour l'entraînement et 20 % pour la validation. Le jeu d'entraînement est l'entrée du modèle ML utilisée pour identifier le meilleur algorithme possible

pour le modèle. Une fois le modèle créé, vous utilisez le jeu de validation pour valider la précision du modèle.

```
CREATE TABLE training_data AS
SELECT
    trip_count,
    trip_hour,
    trip_day,
    trip_month,
    trip_year,
    trip_quarter,
    trip_month_week,
    trip_week_day,
    temp_c,
    precip_amount_mm,
    is_holiday
FROM
    trip_data
WHERE
    serial_number > (
        SELECT
            COUNT(1) * 0.2
        FROM
            trip_data
    );

CREATE TABLE validation_data AS
SELECT
    trip_count,
    trip_hour,
    trip_day,
    trip_month,
    trip_year,
    trip_quarter,
    trip_month_week,
    trip_week_day,
    temp_c,
    precip_amount_mm,
    is_holiday,
    trip_time
FROM
    trip_data
WHERE
```

```
    serial_number <= (
        SELECT
            COUNT(1) * 0.2
        FROM
            trip_data
    );
```

2. La requête suivante crée un modèle de régression pour prédire la valeur `trip_count` pour n'importe quelles date et heure d'entrée. Dans l'exemple suivant, remplacez amzn-s3-demo-bucket par votre propre compartiment S3.

```
CREATE MODEL predict_rental_count
FROM
    training_data TARGET trip_count FUNCTION predict_rental_count
    IAM_ROLE default
    PROBLEM_TYPE regression
    OBJECTIVE 'mse'
    SETTINGS (
        s3_bucket 'amzn-s3-demo-bucket',
        s3_garbage_collect off,
        max_runtime 5000
    );
```

Étape 3 : valider le modèle

1. Utilisez la requête suivante pour générer des aspects du modèle et recherchez la métrique d'écart quadratique moyen dans la sortie. L'écart quadratique moyen est une métrique de précision standard des problèmes de régression.

```
show model predict_rental_count;
```

2. Exécutez les requêtes de prédiction suivantes par rapport aux données de validation pour comparer le nombre prédit de trajets au nombre réel de trajets.

```
SELECT
    trip_time,
    actual_count,
    predicted_count,
    (actual_count - predicted_count) difference
FROM
    (
```

```
        SELECT
            trip_time,
            trip_count AS actual_count,
            PREDICT_RENTAL_COUNT (
                trip_hour,
                trip_day,
                trip_month,
                trip_year,
                trip_quarter,
                trip_month_week,
                trip_week_day,
                temp_c,
                precip_amount_mm,
                is_holiday
            ) predicted_count
        FROM
            validation_data
    )
LIMIT
    5;
```

3. La requête suivante calcule l'écart quadratique moyen et l'écart-type en fonction de vos données de validation. Vous utilisez l'écart quadratique moyen et l'écart-type pour mesurer la distance entre la cible numérique prédite et la réponse numérique réelle. Un bon modèle présente un faible score pour les deux métriques. La requête suivante renvoie la valeur des deux métriques.

```
SELECT
    ROUND(
        AVG(POWER((actual_count - predicted_count), 2)),
        2
    ) mse,
    ROUND(
        SQRT(AVG(POWER((actual_count - predicted_count), 2))),
        2
    ) rmse
FROM
    (
        SELECT
            trip_time,
            trip_count AS actual_count,
            PREDICT_RENTAL_COUNT (
                trip_hour,
                trip_day,
```

```
                trip_month,
                trip_year,
                trip_quarter,
                trip_month_week,
                trip_week_day,
                temp_c,
                precip_amount_mm,
                is_holiday
            ) predicted_count
        FROM
            validation_data
    );
```

4. La requête suivante calcule le pourcentage d'erreur dans le nombre de trajets pour chaque heure de trajet le 01/01/2017. La requête ordonne les heures de trajet de l'heure avec le pourcentage d'erreur le plus bas jusqu'à l'heure avec le pourcentage d'erreur le plus haut.

```
SELECT
    trip_time,
    CAST(ABS(((actual_count - predicted_count) / actual_count)) * 100 AS DECIMAL
 (7,2)) AS pct_error
FROM
    (
        SELECT
            trip_time,
            trip_count AS actual_count,
            PREDICT_RENTAL_COUNT (
                trip_hour,
                trip_day,
                trip_month,
                trip_year,
                trip_quarter,
                trip_month_week,
                trip_week_day,
                temp_c,
                precip_amount_mm,
                is_holiday
            ) predicted_count
        FROM
            validation_data
    )
WHERE
   trip_time LIKE '2017-01-01 %%:%%:%%'
```

```
ORDER BY
   2 ASC;
```

Rubriques en relation

Pour plus d'informations sur Amazon Redshift ML, consultez la documentation suivante :

- Coûts d'utilisation d'Amazon Redshift ML
- Opération CREATE MODEL
- Fonction EXPLAIN_MODEL

Pour plus d'informations sur le machine learning, consultez la documentation suivante :

- Présentation du machine learning
- Machine learning pour les novices et les experts
- En quoi consistent l'équité et l'explicabilité des modèles pour les prédictions de machine learning ?

Tutoriel : Création de modèles de régression avec apprentissage linéaire

Dans ce tutoriel, vous créez un modèle d'apprentissage linéaire avec des données d'Amazon S3 et vous exécutez des requêtes de prédiction avec ce modèle à l'aide d'Amazon Redshift ML. L'algorithme d'apprentissage linéaire basé sur l' SageMaker IA résout les problèmes de régression ou de classification multi-classes. Pour en savoir plus sur les problèmes de régression et de classification multiclasse, consultez la section Types de problèmes liés aux paradigmes d'apprentissage automatique dans le manuel Amazon SageMaker AI Developer Guide. Dans ce tutoriel, vous résolvez un problème de régression. L'algorithme d'apprentissage linéaire entraîne de nombreux modèles en parallèle et détermine automatiquement le modèle le plus optimisé. Vous utilisez l'opération CREATE MODEL dans Amazon Redshift, qui crée votre modèle d'apprenant linéaire à l'aide de l' SageMaker IA et envoie une fonction de prédiction à Amazon Redshift. Pour plus d'informations sur l'algorithme d'apprentissage linéaire, consultez la section Algorithme d'apprentissage linéaire dans le manuel Amazon SageMaker AI Developer Guide.

Vous pouvez utiliser une commande CREATE MODEL pour exporter des données d'entraînement, entraîner un modèle, importer le modèle et préparer une fonction de prédiction Amazon Redshift. Utilisez l'opération CREATE MODEL pour spécifier les données d'entraînement sous la forme d'une table ou d'une instruction SELECT.

Les modèles d'apprentissage linéaire optimisent les objectifs continus ou discrets. Les objectifs continus sont utilisés pour la régression, tandis que les variables discrètes sont utilisées pour la classification. Certaines méthodes fournissent une solution pour les seuls objectifs continus, comme la méthode de régression. L'algorithme d'apprentissage linéaire fournit une hausse de la vitesse par rapport aux techniques naïves d'optimisation des hyperparamètres, telles que la technique naïve bayésienne. Une technique naïve d'optimisation suppose que chaque variable d'entrée est indépendante. Pour utiliser l'algorithme d'apprentissage linéaire, vous devez fournir des colonnes représentant les dimensions des entrées et des lignes représentant les observations. Pour plus d'informations sur l'algorithme d'apprentissage linéaire, consultez l'algorithme d'apprentissage linéaire dans le manuel Amazon SageMaker AI Developer Guide.

Dans ce tutoriel, vous créez un modèle d'apprentissage linéaire qui prédit l'âge des ormeaux. Vous utilisez la commande CREATE MODEL dans le jeu de données Abalone Data Set pour déterminer la relation entre les mesures physiques d'un ormeau. Ensuite, vous utilisez ce modèle pour déterminer l'âge des ormeaux.

Exemples de cas d'utilisation

Vous pouvez résoudre d'autres problèmes de régression avec l'apprentissage linéaire et Amazon Redshift ML, tels que la prédiction du prix d'une maison. Vous pouvez également utiliser Redshift ML pour prédire le nombre de personnes qui utiliseront le service de location de vélos d'une ville.

Tâches

- Prérequis
- Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift
- Étape 2 : Créer le modèle de machine learning
- Étape 3 : valider le modèle

Prérequis

Pour effectuer ce tutoriel, vous devez suivre la procédure Configuration administrative pour Amazon Redshift ML.

Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift

Utilisez l'éditeur de requête v2 Amazon Redshift pour exécuter les requêtes suivantes. Ces requêtes chargent les données d'exemple dans Redshift et divisent les données en un jeu d'entraînement et un jeu de validation.

1. La requête suivante crée la table `abalone_dataset`.

```
CREATE TABLE abalone_dataset (
    id INT IDENTITY(1, 1),
    Sex CHAR(1),
    Length float,
    Diameter float,
    Height float,
    Whole float,
    Shucked float,
    Viscera float,
    Shell float,
    Rings integer
);
```

2. La requête suivante copie les données d'exemple à partir du jeu de données Abalone Data Set dans Amazon S3, dans la table `abalone_dataset` que vous avez créée précédemment dans Amazon Redshift.

```
COPY abalone_dataset
FROM
    's3://redshift-ml-multiclass/abalone.csv' REGION 'us-east-1' IAM_ROLE default CSV
 IGNOREHEADER 1 NULL AS 'NULL';
```

3. En divisant manuellement les données, vous serez en mesure de vérifier la précision du modèle en allouant un jeu de prédiction supplémentaire. La requête suivante divise les données en deux ensembles. La table `abalone_training` est destinée à l'entraînement et la table `abalone_validation` à la validation.

```
CREATE TABLE abalone_training as
SELECT
    *
FROM
    abalone_dataset
WHERE
    mod(id, 10) < 8;

CREATE TABLE abalone_validation as
SELECT
    *
FROM
    abalone_dataset
```

```
WHERE
    mod(id, 10) >= 8;
```

Étape 2 : Créer le modèle de machine learning

Dans cette étape, vous utilisez l'instruction CREATE MODEL pour créer votre modèle de machine learning avec l'algorithme d'apprentissage linéaire.

La requête suivante crée le modèle d'apprentissage linéaire avec l'opération CREATE MODEL à l'aide de votre compartiment S3. Remplacez amzn-s3-demo-bucket par votre propre compartiment S3.

```
CREATE MODEL model_abalone_ring_prediction
FROM
    (
        SELECT
            Sex,
            Length,
            Diameter,
            Height,
            Whole,
            Shucked,
            Viscera,
            Shell,
            Rings AS target_label
        FROM
            abalone_training
    ) TARGET target_label FUNCTION f_abalone_ring_prediction IAM_ROLE default
 MODEL_TYPE LINEAR_LEARNER PROBLEM_TYPE REGRESSION OBJECTIVE 'MSE' SETTINGS (
        S3_BUCKET 'amzn-s3-demo-bucket',
        MAX_RUNTIME 15000
    );
```

Afficher l'état de l'entraînement du modèle (facultatif)

Vous pouvez utiliser la commande SHOW MODEL pour savoir quand votre modèle sera prêt.

Utilisez la requête suivante pour surveiller la progression de l'entraînement du modèle.

```
SHOW MODEL model_abalone_ring_prediction;
```

Lorsque le modèle est prêt, la sortie de l'opération précédente doit ressembler à celle présentée dans l'exemple suivant. Notez que la sortie fournit la métrique `validation:mse`, qui est l'écart quadratique moyen. Vous utiliserez l'écart quadratique moyen pour valider la précision du modèle à l'étape suivante.

```
+--------------------------
+-----------------------------------------------------------------------------------------------
+
|        Model Name        |
 model_abalone_ring_prediction                                    |
+--------------------------
+-----------------------------------------------------------------------------------------------
+
| Schema Name              | public
                                        |
| Owner                    | awsuser
                                        |
| Creation Time            | Thu, 30.06.2022 18:00:10
                                        |
| Model State              | READY
                                        |
| validation:mse           |
                                 4.168633 |
| Estimated Cost           |
                                 4.291608 |
|                          |
                                        |
| TRAINING DATA:           |
                                        |
| Query                    | SELECT SEX , LENGTH , DIAMETER , HEIGHT , WHOLE ,
 SHUCKED , VISCERA , SHELL, RINGS AS TARGET_LABEL |
|                          | FROM ABALONE_TRAINING
                                        |
| Target Column            | TARGET_LABEL
                                        |
|                          |
                                        |
| PARAMETERS:              |
                                        |
| Model Type               | linear_learner
                                        |
| Problem Type             | Regression
                                        |
```

```
| Objective                | MSE
                                    |
| AutoML Job Name          | redshiftml-20220630180010947843
                                    |
| Function Name            | f_abalone_ring_prediction
                                    |
| Function Parameters      | sex length diameter height whole shucked viscera shell
                                    |
| Function Parameter Types | bpchar float8 float8 float8 float8 float8 float8 float8
                                    |
| IAM Role                 | default-aws-iam-role
                                    |
| S3 Bucket                | amzn-s3-demo-bucket
                                    |
| Max Runtime              |
                                15000 |
+--------------------------
+-----------------------------------------------------------------------------------------------------
+
```

Étape 3 : valider le modèle

1. La requête de prédiction suivante valide la précision du modèle sur le jeu de données `abalone_validation` en calculant l'écart quadratique moyen et l'écart-type.

```
SELECT
    ROUND(AVG(POWER((tgt_label - predicted), 2)), 2) mse,
    ROUND(SQRT(AVG(POWER((tgt_label - predicted), 2))), 2) rmse
FROM
    (
        SELECT
            Sex,
            Length,
            Diameter,
            Height,
            Whole,
            Shucked,
            Viscera,
            Shell,
            Rings AS tgt_label,
            f_abalone_ring_prediction(
                Sex,
                Length,
```

```
                Diameter,
                Height,
                Whole,
                Shucked,
                Viscera,
                Shell
            ) AS predicted,
            CASE
                WHEN tgt_label = predicted then 1
                ELSE 0
            END AS match,
            CASE
                WHEN tgt_label <> predicted then 1
                ELSE 0
            END AS nonmatch
        FROM
            abalone_validation
    ) t1;
```

La sortie de la requête précédente doit ressembler à celle de l'exemple suivant. La valeur de la métrique d'écart quadratique moyen doit être similaire à la métrique `validation:mse` affichée par la sortie de l'opération SHOW MODEL.

```
+-----+--------------------+
| mse |        rmse        |
+-----+--------------------+
| 5.1 | 2.2600000000000002 |
+-----+--------------------+
```

2. Utilisez la requête suivante pour exécuter l'opération EXPLAIN_MODEL sur votre fonction de prédiction. Cette opération renvoie un rapport d'explicabilité du modèle. Pour plus d'informations sur l'opération EXPLAIN_MODEL, consultez Fonction EXPLAIN_MODEL dans le Guide du développeur de base de données Amazon Redshift.

```
SELECT
    EXPLAIN_MODEL ('model_abalone_ring_prediction');
```

Les informations suivantes sont un exemple du rapport d'explicabilité de modèle produit par l'opération EXPLAIN_MODEL précédente. Les valeurs de chacune des entrées sont des valeurs de Shapley. Les valeurs de Shapley représentent l'effet de chaque entrée sur la prédiction de votre

modèle, les entrées de valeur supérieure ayant plus d'impact sur la prédiction. Dans cet exemple, les entrées à valeurs élevées ont plus d'impact sur la prédiction de l'âge des ormeaux.

```
{
    "explanations": {
        "kernel_shap": {
            "label0": {
                "expected_value" :10.290688514709473,
                "global_shap_values": {
                    "diameter" :0.6856910187882492,
                    "height" :0.4415323937124035,
                    "length" :0.21507476107609084,
                    "sex" :0.448611774505744,
                    "shell" :1.70426496893776,
                    "shucked" :2.1181392924386994,
                    "viscera" :0.342220754059912,
                    "whole" :0.6711906974084011
                }
            }
        }
    },
    "version" :"1.0"
};
```

3. Utilisez la requête suivante pour calculer le pourcentage de prédictions correctes que le modèle effectue à propos des ormeaux qui ne sont pas encore matures. Les ormeaux encore immatures ont 10 anneaux ou moins, et une prédiction correcte est précise à moins d'un anneau du nombre réel d'anneaux.

```
SELECT
    TRUNC(
        SUM(
            CASE
                WHEN ROUND(
                    f_abalone_ring_prediction(
                        Sex,
                        Length,
                        Diameter,
                        Height,
                        Whole,
                        Shucked,
                        Viscera,
```

```
                        Shell
                    ),
                    0
                ) BETWEEN Rings - 1
                AND Rings + 1 THEN 1
                ELSE 0
            END
        ) / CAST(COUNT(SHELL) AS FLOAT),
        4
    ) AS prediction_pct
FROM
    abalone_validation
WHERE
    Rings <= 10;
```

Rubriques en relation

Pour plus d'informations sur Amazon Redshift ML, consultez la documentation suivante :

- Coûts d'utilisation d'Amazon Redshift ML
- Opération CREATE MODEL
- Fonction EXPLAIN_MODEL

Pour plus d'informations sur le machine learning, consultez la documentation suivante :

- Présentation du machine learning
- Machine learning pour les novices et les experts
- En quoi consistent l'équité et l'explicabilité des modèles pour les prédictions de machine learning ?

Tutoriel : Création de modèles de classification multiclasse avec apprentissage linéaire

Dans ce tutoriel, vous créez un modèle d'apprentissage linéaire avec des données d'Amazon S3, puis vous exécutez des requêtes de prédiction avec le modèle en utilisant Amazon Redshift ML. L'algorithme d'apprentissage linéaire basé sur l' SageMaker IA résout les problèmes de régression ou de classification. Pour en savoir plus sur les problèmes de régression et de classification multiclasse, consultez la section Types de problèmes liés aux paradigmes d'apprentissage automatique dans le

manuel Amazon SageMaker AI Developer Guide. Dans ce tutoriel, vous résolvez un problème de classification multiclasse. L'algorithme d'apprentissage linéaire entraîne de nombreux modèles en parallèle et détermine automatiquement le modèle le plus optimisé. Vous utilisez l'opération CREATE MODEL dans Amazon Redshift, qui crée votre modèle d'apprenant linéaire à l'aide de l' SageMaker IA et envoie la fonction de prédiction à Amazon Redshift. Pour plus d'informations sur l'algorithme d'apprentissage linéaire, consultez l'algorithme d'apprentissage linéaire dans le manuel Amazon SageMaker AI Developer Guide.

Vous pouvez utiliser une commande CREATE MODEL pour exporter des données d'entraînement, entraîner un modèle, importer le modèle et préparer une fonction de prédiction Amazon Redshift. Utilisez l'opération CREATE MODEL pour spécifier les données d'entraînement sous la forme d'une table ou d'une instruction SELECT.

Les modèles d'apprentissage linéaire optimisent les objectifs continus ou discrets. Les objectifs continus sont utilisés pour la régression, tandis que les variables discrètes sont utilisées pour la classification. Certaines méthodes fournissent une solution pour les seuls objectifs continus, comme une méthode de régression. L'algorithme d'apprentissage linéaire fournit une hausse de la vitesse par rapport aux techniques naïves d'optimisation des hyperparamètres, telles que la technique naïve bayésienne. Une technique naïve d'optimisation suppose que chaque variable d'entrée est indépendante. L'algorithme d'apprentissage linéaire entraîne de nombreux modèles en parallèle et sélectionne le modèle le plus optimisé. Un algorithme similaire combine XGBoost les estimations d'un ensemble de modèles plus simples et plus faibles pour faire des prédictions. Pour en savoir plus XGBoost, consultez XGBoost l'algorithme dans le manuel Amazon SageMaker AI Developer Guide.

Pour utiliser l'algorithme d'apprentissage linéaire, vous devez fournir des colonnes représentant les dimensions des entrées et des lignes représentant les observations. Pour plus d'informations sur l'algorithme d'apprentissage linéaire, consultez l'algorithme d'apprentissage linéaire dans le manuel Amazon SageMaker AI Developer Guide.

Dans ce tutoriel, vous créez un modèle d'apprentissage linéaire qui prédit les types de couverture pour une zone donnée. Vous utilisez la commande CREATE MODEL sur le jeu de données Covertype Data Set sur le site Machine Learning Repository de l'UCI. Ensuite, vous utilisez la fonction de prédiction créée par la commande pour déterminer les types de couverture dans une aire de nature sauvage. Un type de couverture forestière est généralement un type d'arbre. Les entrées que Redshift ML utilisera pour créer le modèle incluent le type de sol, la distance aux routes et la désignation des aires de nature sauvage. Pour plus d'informations sur le jeu de données, consultez Covertype Data Set sur le site Machine Learning Repository de l'UCI.

Exemples de cas d'utilisation

Vous pouvez résoudre d'autres problèmes de classification multiclasse avec apprentissage linéaire à l'aide d'Amazon Redshift ML, tels que la prédiction de l'espèce d'une plante à partir d'une image. Vous pouvez également prédire la quantité d'un produit qu'un client achètera.

Tâches

- Prérequis
- Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift
- Étape 2 : Créer le modèle de machine learning
- Étape 3 : valider le modèle

Prérequis

Pour effectuer ce tutoriel, vous devez suivre la procédure Configuration administrative pour Amazon Redshift ML.

Étape 1 : charger les données d'Amazon S3 dans Amazon Redshift

Utilisez l'éditeur de requête v2 Amazon Redshift pour exécuter les requêtes suivantes. Ces requêtes chargent les données d'exemple dans Redshift et divisent les données en un jeu d'entraînement et un jeu de validation.

1. La requête suivante crée la table `covertype_data`.

```
CREATE TABLE public.covertype_data (
    elevation bigint ENCODE az64,
    aspect bigint ENCODE az64,
    slope bigint ENCODE az64,
    horizontal_distance_to_hydrology bigint ENCODE az64,
    vertical_distance_to_hydrology bigint ENCODE az64,
    horizontal_distance_to_roadways bigint ENCODE az64,
    hillshade_9am bigint ENCODE az64,
    hillshade_noon bigint ENCODE az64,
    hillshade_3pm bigint ENCODE az64,
    horizontal_distance_to_fire_points bigint ENCODE az64,
    wilderness_area1 bigint ENCODE az64,
    wilderness_area2 bigint ENCODE az64,
    wilderness_area3 bigint ENCODE az64,
    wilderness_area4 bigint ENCODE az64,
```

```
    soil_type1 bigint ENCODE az64,
    soil_type2 bigint ENCODE az64,
    soil_type3 bigint ENCODE az64,
    soil_type4 bigint ENCODE az64,
    soil_type5 bigint ENCODE az64,
    soil_type6 bigint ENCODE az64,
    soil_type7 bigint ENCODE az64,
    soil_type8 bigint ENCODE az64,
    soil_type9 bigint ENCODE az64,
    soil_type10 bigint ENCODE az64,
    soil_type11 bigint ENCODE az64,
    soil_type12 bigint ENCODE az64,
    soil_type13 bigint ENCODE az64,
    soil_type14 bigint ENCODE az64,
    soil_type15 bigint ENCODE az64,
    soil_type16 bigint ENCODE az64,
    soil_type17 bigint ENCODE az64,
    soil_type18 bigint ENCODE az64,
    soil_type19 bigint ENCODE az64,
    soil_type20 bigint ENCODE az64,
    soil_type21 bigint ENCODE az64,
    soil_type22 bigint ENCODE az64,
    soil_type23 bigint ENCODE az64,
    soil_type24 bigint ENCODE az64,
    soil_type25 bigint ENCODE az64,
    soil_type26 bigint ENCODE az64,
    soil_type27 bigint ENCODE az64,
    soil_type28 bigint ENCODE az64,
    soil_type29 bigint ENCODE az64,
    soil_type30 bigint ENCODE az64,
    soil_type31 bigint ENCODE az64,
    soil_type32 bigint ENCODE az64,
    soil_type33 bigint ENCODE az64,
    soil_type34 bigint ENCODE az64,
    soil_type35 bigint ENCODE az64,
    soil_type36 bigint ENCODE az64,
    soil_type37 bigint ENCODE az64,
    soil_type38 bigint ENCODE az64,
    soil_type39 bigint ENCODE az64,
    soil_type40 bigint ENCODE az64,
    cover_type bigint ENCODE az64
) DISTSTYLE AUTO;
```

2. La requête suivante copie les données d'exemple à partir du jeu de données Covertype Data Set dans Amazon S3, dans la table `covertype_data` que vous avez créée précédemment dans Amazon Redshift.

```
COPY public.covertype_data
FROM
    's3://redshift-ml-multiclass/covtype.data.gz' IAM_ROLE DEFAULT gzip DELIMITER ','
 REGION 'us-east-1';
```

3. En divisant manuellement les données, vous serez en mesure de vérifier la précision du modèle en allouant un jeu de test supplémentaire. La requête suivante divise les données en trois ensembles. La table `covertype_training` est destinée à l'entraînement, la table `covertype_validation` à la validation, et la table `covertype_test` au test du modèle. Vous utiliserez le jeu d'entraînement pour entraîner votre modèle et le jeu de validation pour valider le développement du modèle. Ensuite, vous utilisez le jeu de test pour tester les performances du modèle et voir si le modèle présente un surajustement ou un sous-ajustement du jeu de données.

```
CREATE TABLE public.covertype_data_prep AS
SELECT
    a.*,
    CAST (random() * 100 AS int) AS data_group_id
FROM
    public.covertype_data a;

--training dataset
 CREATE TABLE public.covertype_training as
SELECT
    *
FROM
    public.covertype_data_prep
WHERE
    data_group_id < 80;

--validation dataset
 CREATE TABLE public.covertype_validation AS
SELECT
    *
FROM
    public.covertype_data_prep
WHERE
    data_group_id BETWEEN 80
    AND 89;
```

```
--test dataset
 CREATE TABLE public.covertype_test AS
SELECT
    *
FROM
    public.covertype_data_prep
WHERE
    data_group_id > 89;
```

Étape 2 : Créer le modèle de machine learning

Dans cette étape, vous utilisez l'instruction CREATE MODEL pour créer votre modèle de machine learning avec l'algorithme d'apprentissage linéaire.

La requête suivante crée le modèle d'apprentissage linéaire avec l'opération CREATE MODEL à l'aide de votre compartiment S3. Remplacez amzn-s3-demo-bucket par votre propre compartiment S3.

```
CREATE MODEL forest_cover_type_model
FROM
    (
        SELECT
            Elevation,
            Aspect,
            Slope,
            Horizontal_distance_to_hydrology,
            Vertical_distance_to_hydrology,
            Horizontal_distance_to_roadways,
            HIllshade_9am,
            Hillshade_noon,
            Hillshade_3pm,
            Horizontal_Distance_To_Fire_Points,
            Wilderness_Area1,
            Wilderness_Area2,
            Wilderness_Area3,
            Wilderness_Area4,
            soil_type1,
            Soil_Type2,
            Soil_Type3,
            Soil_Type4,
            Soil_Type5,
```

```
            Soil_Type6,
            Soil_Type7,
            Soil_Type8,
            Soil_Type9,
            Soil_Type10,
            Soil_Type11,
            Soil_Type12,
            Soil_Type13,
            Soil_Type14,
            Soil_Type15,
            Soil_Type16,
            Soil_Type17,
            Soil_Type18,
            Soil_Type19,
            Soil_Type20,
            Soil_Type21,
            Soil_Type22,
            Soil_Type23,
            Soil_Type24,
            Soil_Type25,
            Soil_Type26,
            Soil_Type27,
            Soil_Type28,
            Soil_Type29,
            Soil_Type30,
            Soil_Type31,
            Soil_Type32,
            Soil_Type33,
            Soil_Type34,
            Soil_Type36,
            Soil_Type37,
            Soil_Type38,
            Soil_Type39,
            Soil_Type40,
            Cover_type
        from
            public.covertype_training
    ) TARGET cover_type FUNCTION predict_cover_type IAM_ROLE default MODEL_TYPE
LINEAR_LEARNER PROBLEM_TYPE MULTICLASS_CLASSIFICATION OBJECTIVE 'Accuracy' SETTINGS (
        S3_BUCKET 'amzn-s3-demo-bucket',
        S3_GARBAGE_COLLECT OFF,
        MAX_RUNTIME 15000
    );
```

Afficher l'état de l'entraînement du modèle (facultatif)

Vous pouvez utiliser la commande SHOW MODEL pour savoir quand votre modèle sera prêt.

Utilisez la requête suivante pour surveiller la progression de l'entraînement du modèle.

```
SHOW MODEL forest_cover_type_model;
```

Lorsque le modèle est prêt, la sortie de l'opération précédente doit ressembler à celle présentée dans l'exemple suivant. Notez que la sortie fournit la métrique `validation:multiclass_accuracy`, que vous pouvez voir sur le côté droit de l'exemple suivant. La précision multiclasse mesure le pourcentage des points de données correctement classés par le modèle. Vous utiliserez la précision multiclasse pour valider la précision du modèle à l'étape suivante.

```
+--------------------------------
+-------------------------------------------------------------------------------------------------
+
|              Key               |




 Value




                                                                  |
+--------------------------------
+-------------------------------------------------------------------------------------------------
+
| Model Name                     | forest_cover_type_model






                                                                  |
| Schema Name                    | public
```

```
                                                    |
| Owner                          | awsuser

                                                    |
| Creation Time                  | Tue, 12.07.2022 20:24:32

                                                    |
| Model State                    | READY

                                                    |
| validation:multiclass_accuracy |
```

```
                                        0.724952 |
| Estimated Cost                 |

                                        5.341750 |
|                                |

                                                 |
| TRAINING DATA:                 |

                                                 |
| Query                          | SELECT ELEVATION, ASPECT, SLOPE,
 HORIZONTAL_DISTANCE_TO_HYDROLOGY, VERTICAL_DISTANCE_TO_HYDROLOGY,
 HORIZONTAL_DISTANCE_TO_ROADWAYS, HILLSHADE_9AM, HILLSHADE_NOON, HILLSHADE_3PM ,
 HORIZONTAL_DISTANCE_TO_FIRE_POINTS, WILDERNESS_AREA1, WILDERNESS_AREA2,
 WILDERNESS_AREA3, WILDERNESS_AREA4, SOIL_TYPE1, SOIL_TYPE2, SOIL_TYPE3, SOIL_TYPE4,
 SOIL_TYPE5, SOIL_TYPE6, SOIL_TYPE7, SOIL_TYPE8, SOIL_TYPE9, SOIL_TYPE10 , SOIL_TYPE11,
 SOIL_TYPE12 , SOIL_TYPE13 , SOIL_TYPE14, SOIL_TYPE15, SOIL_TYPE16, SOIL_TYPE17,
 SOIL_TYPE18, SOIL_TYPE19, SOIL_TYPE20, SOIL_TYPE21, SOIL_TYPE22, SOIL_TYPE23,
 SOIL_TYPE24, SOIL_TYPE25, SOIL_TYPE26, SOIL_TYPE27, SOIL_TYPE28, SOIL_TYPE29,
 SOIL_TYPE30, SOIL_TYPE31, SOIL_TYPE32, SOIL_TYPE33, SOIL_TYPE34, SOIL_TYPE36,
 SOIL_TYPE37, SOIL_TYPE38, SOIL_TYPE39, SOIL_TYPE40, COVER_TYPE |
```

```
|                                | FROM PUBLIC.COVERTYPE_TRAINING

                                                                 |
| Target Column                  | COVER_TYPE

                                                                 |
|                                |

                                                                 |
| PARAMETERS:                    |

                                                                 |
| Model Type                     | linear_learner
```

```
                                                    |
| Problem Type                    | MulticlassClassification

                                                    |
| Objective                       | Accuracy

                                                    |
| AutoML Job Name                 | redshiftml-20220712202432187659

                                                    |
| Function Name                   | predict_cover_type
```

```
                                                    |
| Function Parameters            | elevation aspect slope
 horizontal_distance_to_hydrology vertical_distance_to_hydrology
 horizontal_distance_to_roadways hillshade_9am hillshade_noon hillshade_3pm
 horizontal_distance_to_fire_points wilderness_area1 wilderness_area2 wilderness_area3
 wilderness_area4 soil_type1 soil_type2 soil_type3 soil_type4 soil_type5 soil_type6
 soil_type7 soil_type8 soil_type9 soil_type10 soil_type11 soil_type12 soil_type13
 soil_type14 soil_type15 soil_type16 soil_type17 soil_type18 soil_type19 soil_type20
 soil_type21 soil_type22 soil_type23 soil_type24 soil_type25 soil_type26 soil_type27
 soil_type28 soil_type29 soil_type30 soil_type31 soil_type32 soil_type33 soil_type34
 soil_type36 soil_type37 soil_type38 soil_type39 soil_type40
                                                   |
| Function Parameter Types       | int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8 int8
 int8 int8 int8 int8 int8 int8 int8 int8 int8
                                                            |
| IAM Role                       | default-aws-iam-role
                                                    |
| S3 Bucket                      | amzn-s3-demo-bucket
                                                    |
```

```
| Max Runtime                    |

                                                
```

```
            Soil_Type2,
            Soil_Type3,
            Soil_Type4,
            Soil_Type5,
            Soil_Type6,
            Soil_Type7,
            Soil_Type8,
            Soil_Type9,
            Soil_Type10,
            Soil_Type11,
            Soil_Type12,
            Soil_Type13,
            Soil_Type14,
            Soil_Type15,
            Soil_Type16,
            Soil_Type17,
            Soil_Type18,
            Soil_Type19,
            Soil_Type20,
            Soil_Type21,
            Soil_Type22,
            Soil_Type23,
            Soil_Type24,
            Soil_Type25,
            Soil_Type26,
            Soil_Type27,
            Soil_Type28,
            Soil_Type29,
            Soil_Type30,
            Soil_Type31,
            Soil_Type32,
            Soil_Type33,
            Soil_Type34,
            Soil_Type36,
            Soil_Type37,
            Soil_Type38,
            Soil_Type39,
            Soil_Type40,
            Cover_type AS actual_cover_type,
            predict_cover_type(
                Elevation,
                Aspect,
                Slope,
                Horizontal_distance_to_hydrology,
```

```
                Vertical_distance_to_hydrology,
                Horizontal_distance_to_roadways,
                HIllshade_9am,
                Hillshade_noon,
                Hillshade_3pm,
                Horizontal_Distance_To_Fire_Points,
                Wilderness_Area1,
                Wilderness_Area2,
                Wilderness_Area3,
                Wilderness_Area4,
                soil_type1,
                Soil_Type2,
                Soil_Type3,
                Soil_Type4,
                Soil_Type5,
                Soil_Type6,
                Soil_Type7,
                Soil_Type8,
                Soil_Type9,
                Soil_Type10,
                Soil_Type11,
                Soil_Type12,
                Soil_Type13,
                Soil_Type14,
                Soil_Type15,
                Soil_Type16,
                Soil_Type17,
                Soil_Type18,
                Soil_Type19,
                Soil_Type20,
                Soil_Type21,
                Soil_Type22,
                Soil_Type23,
                Soil_Type24,
                Soil_Type25,
                Soil_Type26,
                Soil_Type27,
                Soil_Type28,
                Soil_Type29,
                Soil_Type30,
                Soil_Type31,
                Soil_Type32,
                Soil_Type33,
                Soil_Type34,
```

```
                Soil_Type36,
                Soil_Type37,
                Soil_Type38,
                Soil_Type39,
                Soil_Type40
            ) AS predicted_cover_type,
            CASE
                WHEN actual_cover_type = predicted_cover_type THEN 1
                ELSE 0
            END AS match,
            CASE
                WHEN actual_cover_type <> predicted_cover_type THEN 1
                ELSE 0
            END AS nonmatch
        FROM
            public.covertype_validation
    ) t1;
```

La sortie de la requête précédente doit ressembler à celle de l'exemple suivant.
La valeur de la métrique de précision multiclasse doit être similaire à la métrique `validation:multiclass_accuracy` affichée par la sortie de l'opération SHOW MODEL.

```
+-------------------+----------------------+-------------------+--------------+
| predicted_matches | predicted_non_matches | total_predictions | pct_accuracy |
+-------------------+----------------------+-------------------+--------------+
|             41211 |                16324 |             57535 |   0.71627704 |
+-------------------+----------------------+-------------------+--------------+
```

2. La requête suivante prédit le type de couverture le plus courant pour `wilderness_area2`. Ce jeu de données comprend quatre aires de nature sauvage et sept types de couverture. Une aire de nature sauvage peut avoir plusieurs types de couverture.

```
SELECT t1. predicted_cover_type, COUNT(*)
FROM
(
SELECT
   Elevation,
   Aspect,
   Slope,
   Horizontal_distance_to_hydrology,
   Vertical_distance_to_hydrology,
   Horizontal_distance_to_roadways,
```

```
    HIllshade_9am,
    Hillshade_noon,
    Hillshade_3pm ,
    Horizontal_Distance_To_Fire_Points,
    Wilderness_Area1,
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10 ,
    Soil_Type11,
    Soil_Type12 ,
    Soil_Type13 ,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
    Soil_Type23,
    Soil_Type24,
    Soil_Type25,
    Soil_Type26,
    Soil_Type27,
    Soil_Type28,
    Soil_Type29,
    Soil_Type30,
    Soil_Type31,
    Soil_Type32,
    Soil_Type33,
    Soil_Type34,
    Soil_Type36,
    Soil_Type37,
```

```
    Soil_Type38,
    Soil_Type39,
    Soil_Type40,
    predict_cover_type( Elevation,
    Aspect,
    Slope,
    Horizontal_distance_to_hydrology,
    Vertical_distance_to_hydrology,
    Horizontal_distance_to_roadways,
    HIllshade_9am,
    Hillshade_noon,
    Hillshade_3pm ,
    Horizontal_Distance_To_Fire_Points,
    Wilderness_Area1,
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10,
    Soil_Type11,
    Soil_Type12,
    Soil_Type13,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
    Soil_Type23,
    Soil_Type24,
    Soil_Type25,
    Soil_Type26,
    Soil_Type27,
```

```
   Soil_Type28,
   Soil_Type29,
   Soil_Type30,
   Soil_Type31,
   Soil_Type32,
   Soil_Type33,
   Soil_Type34,
   Soil_Type36,
   Soil_Type37,
   Soil_Type38,
   Soil_Type39,
   Soil_Type40) AS predicted_cover_type

FROM public.covertype_test
WHERE wilderness_area2 = 1)
t1
GROUP BY 1;
```

La sortie de l'opération précédente doit ressembler à celle de l'exemple suivant. Cette sortie signifie que le modèle a prédit que la majeure partie de la couverture correspond au type de couverture 1, et qu'une partie de la couverture présente les types de couverture 2 et 7.

```
+----------------------+-------+
| predicted_cover_type | count |
+----------------------+-------+
|                    2 |   564 |
|                    7 |    97 |
|                    1 |  2309 |
+----------------------+-------+
```

3. La requête suivante montre le type de couverture le plus courant dans une aire de nature sauvage individuelle. La requête affiche la quantité de ce type de couverture et l'aire de nature sauvage de ce type de couverture.

```
SELECT t1. predicted_cover_type, COUNT(*), wilderness_area
FROM
(
SELECT
   Elevation,
   Aspect,
   Slope,
   Horizontal_distance_to_hydrology,
```

```
    Vertical_distance_to_hydrology,
    Horizontal_distance_to_roadways,
    HIllshade_9am,
    Hillshade_noon,
    Hillshade_3pm ,
    Horizontal_Distance_To_Fire_Points,
    Wilderness_Area1,
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10 ,
    Soil_Type11,
    Soil_Type12 ,
    Soil_Type13 ,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
    Soil_Type23,
    Soil_Type24,
    Soil_Type25,
    Soil_Type26,
    Soil_Type27,
    Soil_Type28,
    Soil_Type29,
    Soil_Type30,
    Soil_Type31,
    Soil_Type32,
    Soil_Type33,
    Soil_Type34,
```

```
    Soil_Type36,
    Soil_Type37,
    Soil_Type38,
    Soil_Type39,
    Soil_Type40,
    predict_cover_type( Elevation,
    Aspect,
    Slope,
    Horizontal_distance_to_hydrology,
    Vertical_distance_to_hydrology,
    Horizontal_distance_to_roadways,
    HIllshade_9am,
    Hillshade_noon,
    Hillshade_3pm ,
    Horizontal_Distance_To_Fire_Points,
    Wilderness_Area1,
    Wilderness_Area2,
    Wilderness_Area3,
    Wilderness_Area4,
    soil_type1,
    Soil_Type2,
    Soil_Type3,
    Soil_Type4,
    Soil_Type5,
    Soil_Type6,
    Soil_Type7,
    Soil_Type8,
    Soil_Type9,
    Soil_Type10,
    Soil_Type11,
    Soil_Type12,
    Soil_Type13,
    Soil_Type14,
    Soil_Type15,
    Soil_Type16,
    Soil_Type17,
    Soil_Type18,
    Soil_Type19,
    Soil_Type20,
    Soil_Type21,
    Soil_Type22,
    Soil_Type23,
    Soil_Type24,
    Soil_Type25,
```

```
   Soil_Type26,
   Soil_Type27,
   Soil_Type28,
   Soil_Type29,
   Soil_Type30,
   Soil_Type31,
   Soil_Type32,
   Soil_Type33,
   Soil_Type34,
   Soil_Type36,
   Soil_Type37,
   Soil_Type38,
   Soil_Type39,
   Soil_Type40) AS predicted_cover_type,
   CASE WHEN Wilderness_Area1 = 1 THEN 1
        WHEN Wilderness_Area2 = 1 THEN 2
        WHEN Wilderness_Area3 = 1 THEN 3
        WHEN Wilderness_Area4 = 1 THEN 4
        ELSE 0
   END AS wilderness_area

FROM public.covertype_test)
t1
GROUP BY 1, 3
ORDER BY 2 DESC
LIMIT 1;
```

La sortie de l'opération précédente doit ressembler à celle de l'exemple suivant.

```
+----------------------+-------+-----------------+
| predicted_cover_type | count | wilderness_area |
+----------------------+-------+-----------------+
|                    2 | 15738 |               1 |
+----------------------+-------+-----------------+
```

Rubriques en relation

Pour plus d'informations sur Amazon Redshift ML, consultez la documentation suivante :

- Coûts d'utilisation d'Amazon Redshift ML
- Opération CREATE MODEL

- Fonction EXPLAIN_MODEL

Pour plus d'informations sur le machine learning, consultez la documentation suivante :

- Présentation du machine learning
- Machine learning pour les novices et les experts
- En quoi consistent l'équité et l'explicabilité des modèles pour les prédictions de machine learning ?

Intégration d'Amazon Redshift ML à Amazon Bedrock

Cette section explique comment utiliser l'intégration d'Amazon Redshift ML avec Amazon Bedrock. Grâce à cette fonctionnalité, vous pouvez invoquer un modèle Amazon Bedrock à l'aide de SQL, et vous pouvez utiliser vos données issues d'un entrepôt de données Amazon Redshift pour créer des applications d'IA génératives telles que la génération de texte, l'analyse des sentiments ou la traduction.

Rubriques

- Création ou mise à jour d'un rôle IAM pour l'intégration d'Amazon Redshift ML à Amazon Bedrock
- Création d'un modèle externe pour l'intégration d'Amazon Redshift ML à Amazon Bedrock
- Utilisation d'un modèle externe pour l'intégration d'Amazon Redshift ML à Amazon Bedrock
- Ingénierie rapide pour l'intégration d'Amazon Redshift ML à Amazon Bedrock

Création ou mise à jour d'un rôle IAM pour l'intégration d'Amazon Redshift ML à Amazon Bedrock

Cette section explique comment créer un rôle IAM à utiliser dans le cadre de l'intégration d'Amazon Redshift ML à Amazon Bedrock.

Ajoutez la politique suivante au rôle IAM que vous utilisez dans le cadre de l'intégration d'Amazon Redshift ML à Amazon Bedrock :

- `AmazonBedrockFullAccess`

Pour permettre à Amazon Redshift d'assumer un rôle d'interaction avec d'autres services, ajoutez la politique de confiance suivante au rôle IAM :

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

Si le cluster ou l'espace de noms se trouve dans un VPC, suivez les instructions figurant dans. Cluster et configuration de la mise en place pour l'administration Amazon Redshift ML

Si vous avez besoin d'une politique plus restrictive, vous pouvez en créer une qui inclut uniquement les autorisations Amazon Bedrock spécifiées dans les pages suivantes :

- Cluster et configuration de la mise en place pour l'administration Amazon Redshift ML
- Autorisations requises pour utiliser le machine learning (ML) Amazon Redshift

Pour plus d'informations sur la création d'un rôle IAM, consultez la section Création d'un rôle IAM dans le guide de l'AWS Identity and Access Management utilisateur.

Création d'un modèle externe pour l'intégration d'Amazon Redshift ML à Amazon Bedrock

Cette section explique comment créer un modèle externe à utiliser comme interface pour Amazon Bedrock au sein de votre entrepôt de données Amazon Redshift.

Pour appeler un modèle Amazon Bedrock depuis Amazon Redshift, vous devez d'abord exécuter `CREATE EXTERNAL MODEL` la commande. Cette commande crée un objet modèle externe dans la base de données et une fonction utilisateur associée que vous utilisez pour générer du contenu textuel avec Amazon Bedrock.

L'exemple de code suivant montre une `CREATE EXTERNAL MODEL` commande de base :

```
CREATE EXTERNAL MODEL llm_claude
FUNCTION llm_claude_func
IAM_ROLE '<IAM role arn>'
MODEL_TYPE BEDROCK
SETTINGS (
   MODEL_ID 'anthropic.claude-v2:1',
   PROMPT 'Summarize the following text:');
```

La `CREATE EXTERNAL MODEL` commande dispose d'une interface unifiée et cohérente avec Amazon Bedrock for all Foundation Models (FM) prenant en charge les messages. Il s'agit de l'option par défaut lorsque vous utilisez la `CREATE EXTERNAL MODEL` commande ou lorsque vous spécifiez explicitement le type de demande à être`UNIFIED`. Pour plus d'informations, consultez la documentation de l'API Converse dans la documentation de l'API Amazon Bedrock.

Si un FM ne prend pas en charge les messages, vous devez régler le `request_type` réglage sur`RAW`. Lorsque vous réglez `request_type` sur`RAW`, vous devez créer la demande envoyée à Amazon Bedrock lorsque vous utilisez la fonction d'inférence en fonction du FM sélectionné.

Le `PROMPT` paramètre de la `CREATE EXTERNAL MODEL` commande est une invite statique. Si vous avez besoin d'une invite dynamique pour votre application, vous devez la spécifier lorsque vous utilisez la fonction d'inférence. Pour plus de détails, voir ce Ingénierie rapide pour l'intégration d'Amazon Redshift ML à Amazon Bedrock qui suit.

Pour plus d'informations sur l'`CREATE EXTERNAL MODEL`instruction, ses paramètres et ses réglages, consultezCRÉER UN MODÈLE EXTERNE.

Utilisation d'un modèle externe pour l'intégration d'Amazon Redshift ML à Amazon Bedrock

Cette section explique comment invoquer un modèle externe pour générer du texte en réponse aux instructions fournies. Pour invoquer un modèle externe, utilisez la fonction d'inférence que vous créez avec`CREATE EXTERNAL MODEL`.

Rubriques

- Inférence avec des modèles de type de UNIFIED demande
- Inférence avec des modèles de type de RAW demande
- Fonctions d'inférence en tant que fonctions de leader uniquement
- Remarques sur l'utilisation de la fonction d'inférence

Inférence avec des modèles de type de **UNIFIED** demande

La fonction d'inférence pour les modèles avec type de requête UNIFIED possède les trois paramètres suivants qui sont transmis à la fonction dans l'ordre :

- Texte de saisie (obligatoire) : ce paramètre spécifie le texte d'entrée qu'Amazon Redshift transmet à Amazon Bedrock.
- Configuration d'inférence et champs de demande de modèle supplémentaires (facultatif) : Amazon Redshift transmet ces paramètres aux paramètres correspondants pour l'API du modèle Converse.

L'exemple de code suivant montre comment utiliser une fonction d'inférence UNIFIED de type :

```
SELECT llm_claude_func(input_text, object('temperature', 0.7, 'maxtokens', 500))
   FROM some_data;
```

Inférence avec des modèles de type de **RAW** demande

La fonction d'inférence pour les modèles avec type de demande ne RAW comporte qu'un seul paramètre de type SUPER de données. La syntaxe de ce paramètre dépend du modèle Amazon Bedrock utilisé.

L'exemple de code suivant montre comment utiliser une fonction d'inférence RAW de type :

```
SELECT llm_titan_func(
    object(
        "inputText", "Summarize the following text: " | input_text,
        "textGenerationConfig", object("temperature", 0.5, "maxTokenCount", 500)
    )
)
FROM some_data;
```

Fonctions d'inférence en tant que fonctions de leader uniquement

Les fonctions d'inférence pour les modèles Amazon Bedrock peuvent être exécutées en tant que fonctions du nœud principal uniquement lorsque la requête qui les utilise ne fait référence à aucune table. Cela peut être utile si vous souhaitez poser rapidement une question à un LLM.

L'exemple de code suivant montre comment utiliser une fonction d'inférence réservée aux leaders :

```
SELECT general_titan_llm_func('Summarize the benefits of LLM on data analytics in 100
 words');
```

Remarques sur l'utilisation de la fonction d'inférence

Notez les points suivants lorsque vous utilisez des fonctions d'inférence avec l'intégration d'Amazon Redshift ML à Amazon Bedrock :

- Les noms des paramètres de tous les modèles Amazon Bedrock distinguent les majuscules et minuscules. Si vos paramètres ne correspondent pas à ceux requis par le modèle, Amazon Bedrock risque de les ignorer discrètement.
- Le débit des requêtes d'inférence est limité par les quotas d'exécution des différents modèles proposés par Amazon Bedrock dans différentes régions. Pour plus d'informations, consultez la section Quotas pour Amazon Bedrock dans le guide de l'utilisateur d'Amazon Bedrock.
- Si vous avez besoin d'un débit garanti et constant, pensez à obtenir un débit provisionné pour le modèle dont vous avez besoin auprès d'Amazon Bedrock. Pour plus d'informations, consultez la section Augmenter la capacité d'invocation des modèles avec le débit provisionné dans Amazon Bedrock dans le guide de l'utilisateur d'Amazon Bedrock.
- Les requêtes d'inférence contenant de grandes quantités de données peuvent faire l'objet d'exceptions de limitation. Cela est dû aux quotas d'exécution limités pour Amazon Bedrock. Amazon Redshift réessaie plusieurs fois les demandes, mais celles-ci peuvent toujours être limitées car le débit des modèles non provisionnés peut être variable.
- Si vous rencontrez des exceptions de limitation provenant d'Amazon Bedrock, par exemple avec de petites quantités de données, vérifiez les quotas dans la section Service Quotas `Too many requests, please wait before trying again.` de votre compte Amazon Bedrock. Vérifiez que le quota appliqué au niveau du compte est au moins identique à la valeur de quota AWS par défaut pour les InvokeModeldemandes relatives au modèle que vous utilisez.

Ingénierie rapide pour l'intégration d'Amazon Redshift ML à Amazon Bedrock

Cette section explique comment utiliser les instructions statiques avec un modèle externe.

Pour utiliser des instructions de préfixe et de suffixe statiques pour votre modèle externe, fournissez-les à l'aide des `SUFFIX` paramètres `PROMPT` et de l'instruction. `CREATE EXTERNAL MODEL` Ces instructions sont ajoutées à chaque requête à l'aide du modèle externe.

L'exemple suivant montre comment ajouter des instructions de préfixe et de suffixe à un modèle externe :

```
CREATE EXTERNAL MODEL llm_claude
FUNCTION llm_claude_func
IAM_ROLE '<IAM role arn>'
MODEL_TYPE BEDROCK
SETTINGS (
   MODEL_ID 'anthropic.claude-v2:1',
   PROMPT 'Summarize the following text:',
   SUFFIX 'Respond in an analytic tone');
```

Pour utiliser des invites dynamiques, vous pouvez les fournir lorsque vous utilisez la fonction d'inférence en les concaténant dans l'entrée de la fonction. L'exemple suivant montre comment utiliser des invites dynamiques avec une fonction d'inférence :

```
SELECT llm_claude_func('Summarize the following review:' | input_text | 'The review
 should have formal tone.')
FROM some_data
```

Réglage des performances des requêtes

Amazon Redshift utilise des requêtes basées sur le langage de requête structuré (SQL) pour interagir avec les données et les objets du système. Le langage de manipulation de données (Data Manipulation Language, DML) constitue le sous-ensemble de SQL que vous utilisez pour afficher, ajouter, modifier et supprimer des données. Le langage de définition de données (Data definition language, DDL) constitue le sous-ensemble de SQL que vous utilisez pour ajouter, modifier et supprimer des objets de base de données comme des tables et des vues.

Une fois votre système configuré, vous utilisez généralement le plus souvent le DML, en particulier la commande SELECT permettant de récupérer et d'afficher des données. Pour écrire des requêtes d'extraction de données efficaces dans Amazon Redshift, familiarisez-vous avec SELECT et appliquez les conseils décrits dans la section Bonnes pratiques Amazon Redshift pour la conception de tables pour optimiser l'efficacité des requêtes.

Pour comprendre comment Amazon Redshift traite les requêtes, consultez les rubriques Traitement des requêtes et Analyse et amélioration des requêtes. Ensuite, vous pouvez appliquer ces informations en liaison avec des outils de diagnostic afin d'identifier et de supprimer les problèmes de performances des requêtes.

Pour identifier et résoudre certains des problèmes les plus courants et les plus graves que vous êtes susceptibles de rencontrer avec les requêtes Amazon Redshift, utilisez la section Résolution des problèmes de requêtes.

Rubriques

- Traitement des requêtes
- Analyse et amélioration des requêtes
- Résolution des problèmes de requêtes

Traitement des requêtes

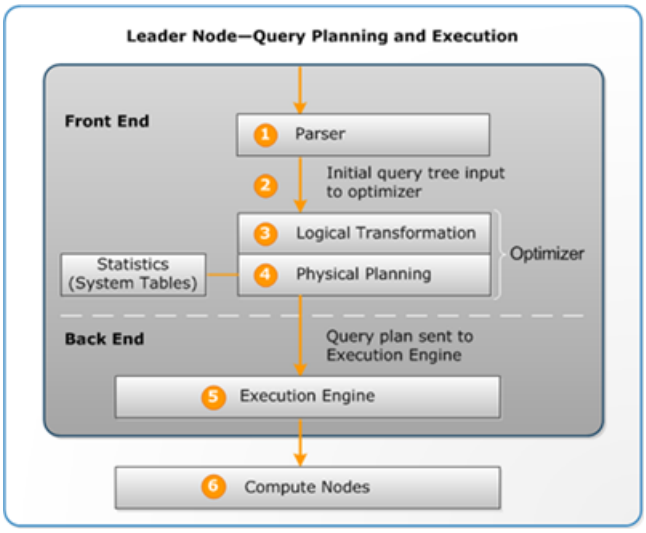
Amazon Redshift achemine une requête SQL soumise à travers l'analyseur et l'optimiseur pour développer un plan de requête. Le moteur d'exécution traduit ensuite le plan de requête en code et envoie celui-ci aux nœuds de calcul à des fins d'exécution.

Rubriques

- Workflow d’exécution et de planification de requête
- Création et interprétation d'un plan de requêtes
- Révision des étapes du plan de requête
- Facteurs affectant la performance des requêtes

Workflow d’exécution et de planification de requête

L’illustration suivante fournit présentation d’ensemble du workflow d’exécution et de planification de la requête.



Le workflow de planification et d’exécution de la requête se déroule comme suit :

1. Le nœud principal reçoit la requête et analyse le SQL.
2. L’analyseur produit un arbre de requête initial qui est une représentation logique de la requête originale. Amazon Redshift saisit ensuite cette arborescence de requêtes dans l’optimiseur de requêtes.
3. L’optimiseur évalue et, si nécessaire, réécrit la requête afin d’optimiser son efficacité. Ce processus entraîne parfois la création de plusieurs requêtes connexes pour en remplacer une seule.
4. L’optimiseur génère un plan de requête (ou plusieurs, si l’étape précédente entraîné la création de plusieurs requêtes) pour que l’exécution s’effectue avec les meilleures performances. Le plan de requête spécifie les options d’exécution telles que les types de jointures, l’ordre des jointures, les options d’agrégation et les exigences de distribution des données.

Vous pouvez utiliser la commande EXPLAIN pour afficher le plan de requête. Le plan de requête est un outil fondamental permettant d'analyser et d'ajuster des requêtes complexes. Pour de plus amples informations, veuillez consulter Création et interprétation d'un plan de requêtes.

5. Le moteur d'exécution traduit le plan de requête en étapes, segments et flux :

   Étape

   Chaque étape est une opération individuelle nécessaire au cours de l'exécution des requêtes. Les étapes peuvent être combinées afin d'autoriser les nœuds de calcul à effectuer une requête, créer une jointure ou une autre opération de base de données.

   Segment

   Combinaison de plusieurs étapes pouvant être effectuées par un processus unique, également la plus petite unité de compilation exécutable par une tranche de nœud de calcul. Une tranche est l'unité de traitement parallèle dans Amazon Redshift. Les segments d'un flux s'exécutent en parallèle.

   Flux

   Ensemble de segments à répartir entre les tranches de nœuds de calcul disponibles.

   Le moteur d'exécution génère du code compilé basé sur les étapes, les segments et les flux. Le code compilé s'exécute plus vite qu'un code interprété et utilise moins de capacité de calcul. Ce code compilé est ensuite diffusé aux nœuds de calcul.

   > **Note**
   >
   > Lors de l'évaluation de vos requêtes, vous devez toujours comparer les durées de la seconde exécution d'une requête, car la première durée d'exécution inclut la surcharge due à la compilation du code. Pour de plus amples informations, veuillez consulter Facteurs affectant la performance des requêtes.

6. Les tranches de nœuds de calcul exécutent les segments de requête en parallèle. Dans le cadre de ce processus, Amazon Redshift tire parti d'une gestion optimisée des communications réseau, de la mémoire et des disques pour transmettre les résultats intermédiaires d'une étape du plan de requête à la suivante. Cela contribue également à accélérer l'exécution des requêtes.

Les étapes 5 et 6 s'effectuent une fois par flux. Le moteur crée les segments exécutables pour un flux et les envoie vers les nœuds de calcul. Lorsque les segments de ce flux sont terminés, le moteur génère les segments pour le prochain flux. Ainsi, le moteur peut analyser ce qui est arrivé dans le flux précédent (par exemple, si les opérations étaient basées sur le disque) pour influencer la génération de segments dans le flux suivant.

Lorsque les nœuds de calcul sont terminés, ils renvoient les résultats de la requête au nœud principal pour le traitement final. Le nœud principal fusionne les données en un seul jeu de résultats et traite tous les tris ou agrégations nécessaires. Le nœud principal renvoie ensuite les résultats au client.

> **Note**
>
> Au besoin, les nœuds de calcul peuvent renvoyer des données au nœud principal au cours de l'exécution des requêtes. Par exemple, si vous avez une sous-requête comportant une clause LIMIT, la limite est appliquée au nœud principal avant que les données soient redistribuées dans le cluster en vue d'un traitement ultérieur.

Création et interprétation d'un plan de requêtes

Vous pouvez utiliser le plan de requête pour obtenir des informations sur les opérations individuelles requises pour exécuter une requête. Avant de travailler avec un plan de requête, nous vous recommandons de comprendre d'abord comment Amazon Redshift gère le traitement des requêtes et la création des plans de requête. Pour de plus amples informations, veuillez consulter Workflow d'exécution et de planification de requête.

Pour créer un plan de requête, exécutez la commande EXPLAIN suivie du texte de la requête réelle. Le plan de requête vous fournit les informations suivantes :

- Les opérations effectuées par le moteur d'exécution, en lisant les résultats de bas en haut.
- Le type d'étape effectué par chaque opération.
- Les tables et les colonnes utilisées dans chaque opération.
- La quantité de données traitée dans chaque opération, en termes de nombre de lignes et de largeur de données en octets.
- Le coût relatif de l'opération. Cost est une mesure qui compare les durées d'exécution relatives des étapes au sein d'un plan. Cost ne fournit pas d'informations précises sur les durées d'exécution ou la consommation de mémoire réelle, ni de comparaison significative des plans d'exécution. Il vous donne une indication des opérations d'une requête qui consomment le plus de ressources.

La commande EXPLAIN n'exécute pas réellement la requête. Elle montre seulement le plan que Amazon Redshift exécutera si la requête est exécutée dans les conditions d'utilisation actuelles. Si vous modifiez le schéma ou les données d'une table et que vous exécutez ANALYSE à nouveau pour mettre à jour les métadonnées statistiques, le plan de requête peut être différent.

La sortie du plan de requête par EXPLAIN est une vue simplifiée et d'ensemble de l'exécution des requêtes. Il n'illustre pas les détails du traitement de requête parallèle. Pour consulter des informations détaillées, vous devez exécuter la requête elle-même, puis obtenir des informations récapitulatives sur la requête dans la vue SVL_QUERY_SUMMARY ou SVL_QUERY_REPORT. Pour plus d'informations sur l'utilisation de ces vues, consultez Analyse du résumé de la requête.

L'exemple suivant illustre la sortie EXPLAIN pour une requête GROUP BY simple sur la table EVENT :

```
explain select eventname, count(*) from event group by eventname;

                            QUERY PLAN
------------------------------------------------------------------
XN HashAggregate  (cost=131.97..133.41 rows=576 width=17)
  ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=17)
```

EXPLAIN renvoie les métriques suivantes pour chaque opération :

Coût

Valeur relative utile pour comparer les opérations au sein d'un plan. Cost se compose de deux valeurs décimales séparées par des deux points, par exemple `cost=131.97..133.41`. La première valeur, dans le cas présent 131.97, fournit le coût relatif du renvoi de la première ligne pour cette opération. La seconde valeur, dans le cas présent 133.41, fournit le coût relatif de l'exécution de l'opération. Les coûts du plan de requête sont cumulatifs au fur et à mesure que vous lisez le plan. Le HashAggregate coût de cet exemple (131.97.. 133.41) inclut le coût du Seq Scan situé en dessous (0,00.. 87,98).

Lignes

Estimation du nombre de lignes à renvoyer. Dans cet exemple, l'analyse devrait renvoyer 8 798 lignes. L' HashAggregate opérateur lui-même est censé renvoyer 576 lignes (une fois que les noms d'événements dupliqués ont été supprimés du jeu de résultats).

> **Note**
>
> L'estimation de lignes repose sur les statistiques disponibles générées par la commande ANALYZE. Si ANALYZE n'a pas été exécutée récemment, l'estimation sera moins fiable.

Largeur

Largeur estimée de la ligne moyenne, en octets. Dans cet exemple, la ligne moyenne devrait avoir une largeur de 17 octets.

Opérateurs EXPLAIN

Cette section décrit brièvement les opérateurs que vous voyez le plus souvent dans la sortie EXPLAIN. Pour obtenir une liste complète des opérateurs, consultez EXPLAIN dans la section Commandes SQL.

Opérateur d'analyse séquentielle

L'opérateur d'analyse séquentiel (Seq Scan) indique une analyse de table. Seq Scan analyse chaque colonne de la tableau de manière séquentielle du début à la fin et évalue les contraintes de requête (dans la clause WHERE) de chaque ligne.

Opérateurs de jointure

Amazon Redshift sélectionne les opérateurs de jointure en fonction de la conception physique des tables jointes, de l'emplacement des données requises pour la jointure et des exigences spécifiques à la requête elle-même.

- Boucle imbriquée

  La jointure la moins optimale, une boucle imbriquée, est utilisée principalement pour les jointures croisées (produits cartésiens) et certaines jointures d'inégalité.
- Hash Join and Hash

  Généralement plus rapide qu'une boucle imbriquée, une jointure par hachage et un hachage sont utilisés pour les jointures internes et les jointures externes gauche et droite. Ces opérateurs sont utilisés lors de la jonction de tables, lorsque les colonnes de jointure ne sont pas des clés de distribution et des clés de tri. L'opérateur de hachage crée la table de hachage pour la table interne

de la jointure ; l'opérateur de jointure par hachage lit la table externe, hache la colonne de jointure et recherche des correspondances dans la table de hachage interne.

- Joindre par fusion

  Généralement la jointure la plus rapide, une jointure par fusion est utilisée pour les jointures internes et externes. La jointure par fusion n'est pas utilisée pour les jointures complètes. Cet opérateur est utilisé lors de la jonction de tables lorsque les colonnes de jointure sont des clés de distribution et des clés de tri, et lorsque moins de 20 % des tables jointes sont non triées. Il lit les deux tables triées dans l'ordre et recherche les lignes correspondantes. Pour afficher le pourcentage de lignes non triées, interrogez la table système SVV_TABLE_INFO.

- Jointure spatiale

  Il s'agit généralement d'une jointure rapide basée sur la proximité des données spatiales, utilisées pour les types de données `GEOMETRY` et `GEOGRAPHY`.

Opérateurs d'agrégation

Le plan de requête utilise les opérateurs suivants dans les requêtes impliquant des fonctions d'agrégation et des opérations GROUP BY.

- Regrouper

  Opérateur de fonctions d'agrégation scalaires telles que AVG et SUM.

- HashAggregate

  Opérateur des fonctions d'agrégation groupées non triées.

- GroupAggregate

  Opérateur des fonctions d'agrégation groupées triées.

Opérateurs de tri

Le plan de requête utilise les opérateurs suivants lorsque les requêtes doivent trier ou fusionner des jeux de résultats.

- Tri

  Evalue la clause ORDER BY et d'autres opérations de tri, telles que les tris requis par les requêtes UNION et les requêtes SELECT DISTINCT de jointure et les fonctions de fenêtrage.

- Fusionner

  Produit des résultats triés finaux selon les résultats triés intermédiaires qui proviennent d’opérations parallèles.

Opérateurs UNION, INTERSECT et EXCEPT

Le plan de requête utilise les opérateurs suivants pour des requêtes impliquant des opérations de jeu avec UNION, INTERSECT et EXCEPT.

- Subquery

  Utilisé pour exécuter des requêtes UNION.

- Hash Intersect Distinct

  Utilisé pour exécuter les requêtes INTERSECT .

- SetOp Sauf

  Utilisé pour exécuter des requêtes EXCEPT (ou MINUS).

Autres opérateurs

Les opérateurs suivants apparaissent également fréquemment dans la sortie EXPLAIN pour les requêtes courantes.

- Unique

  Supprime les doublons des requêtes SELECT DISTINCT et UNION.

- Limite

  Traite la clause LIMIT.

- Fenêtre

  Exécute les fonctions de fenêtrage.

- Result

  Exécute les fonctions scalaires qui n’impliquent pas un accès aux tables.

- Subplan

Utilisé pour certaines sous-requêtes.

- Réseau

  Envoie des résultats intermédiaires au nœud principal en vue d'un traitement ultérieur.

- Materialize

  Enregistre les lignes pour l'entrée des jointures de boucles imbriquées et quelques autres jointures de fusion.

Jointures dans EXPLAIN

L'optimiseur de requête utilise différents types de jointures pour récupérer les données de la table, en fonction de la structure de la requête et des tables sous-jacentes. La sortie EXPLAIN fait référence au type de jointure, aux tables utilisées et à la manière dont les données de la table sont distribuées dans le cluster afin de décrire le traitement de la requête.

Exemples de types de jointures

Les exemples suivants illustrent les différents types de jointures que l'optimiseur de requête peut utiliser. Le type de jointure utilisé dans le plan de requête dépend de la conception physique des tables impliquées.

Exemple : Joindre par hachage deux tables

La requête suivante joint EVENT et CATEGORY sur la colonne CATID. CATID est la clé de tri et de distribution pour CATEGORY, mais pas pour EVENT. Une jointure par hachage est effectuée avec EVENT en tant que table externe et avec CATEGORY en tant que table interne. Etant donné que CATEGORY est la table la plus petite, le planificateur en diffuse une copie aux nœuds de calcul au cours du traitement de la requête à l'aide de DS_BCAST_INNER. Le coût de la jointure de cet exemple représente la majeure partie du coût cumulé du plan.

```
explain select * from category, event where category.catid=event.catid;

                              QUERY PLAN
------------------------------------------------------------------------
 XN Hash Join DS_BCAST_INNER  (cost=0.14..6600286.07 rows=8798 width=84)
   Hash Cond: ("outer".catid = "inner".catid)
   ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=35)
   ->  XN Hash  (cost=0.11..0.11 rows=11 width=49)
```

```
         ->  XN Seq Scan on category  (cost=0.00..0.11 rows=11 width=49)
```

> **Note**
>
> Les retraits alignés pour les opérateurs dans la sortie EXPLAIN indiquent parfois que ces opérations ne dépendent pas l'une de l'autre et peuvent démarrer en parallèle. Dans l'exemple précédent, bien que l'analyse de la table EVENT et l'opération de hachage soient alignées, l'analyse EVENT doit attendre que l'opération de hachage soit entièrement terminée.

Exemple : Joindre par fusion deux tables

La requête suivante utilise également SELECT *, mais elle joint SALES et LISTING sur la colonne LISTID, dans laquelle LISTID a été défini comme clé de tri et de distribution des deux tables. Une jointure par fusion est choisie et aucun redistribution des données n'est requise pour la jointure (DS_DIST_NONE).

```
explain select * from sales, listing where sales.listid = listing.listid;
QUERY PLAN
------------------------------------------------------------------------------
XN Merge Join DS_DIST_NONE  (cost=0.00..6285.93 rows=172456 width=97)
  Merge Cond: ("outer".listid = "inner".listid)
  ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=44)
  ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=53)
```

L'exemple suivant illustre les différents types de jointures au sein de la même requête. Comme dans l'exemple précédent, SALES et LISTING sont jointes par fusion, mais la troisième table, EVENT, doit être jointe par hachage aux résultats de la jointure par fusion. Une fois encore, la jointure par hachage implique un coût de diffusion.

```
explain select * from sales, listing, event
where sales.listid = listing.listid and sales.eventid = event.eventid;
                                 QUERY PLAN
------------------------------------------------------------------------------
XN Hash Join DS_BCAST_INNER  (cost=109.98..3871130276.17 rows=172456 width=132)
  Hash Cond: ("outer".eventid = "inner".eventid)
  ->  XN Merge Join DS_DIST_NONE  (cost=0.00..6285.93 rows=172456 width=97)
        Merge Cond: ("outer".listid = "inner".listid)
        ->  XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=44)
```

```
       ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=53)
  ->  XN Hash  (cost=87.98..87.98 rows=8798 width=35)
       ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=35)
```

Exemple : Joindre, regrouper et trier

La requête suivante exécute une jointure par hachage des tables SALES et EVENT, suivie d'opérations d'agrégation et de tri afin de tenir compte de la fonction SUM groupée et de la clause ORDER BY. L'opérateur Sort initial s'exécute en parallèle sur les nœuds de calcul. Puis, l'opérateur Network envoie les résultats au nœud principal, dans lequel l'opérateur Merge produit les résultats triés finaux.

```
explain select eventname, sum(pricepaid) from sales, event
where sales.eventid=event.eventid group by eventname
order by 2 desc;
                                            QUERY PLAN
------------------------------------------------------------------------------
 XN Merge  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
  Merge Key: sum(sales.pricepaid)
  ->  XN Network  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
        Send to leader
        ->  XN Sort  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
              Sort Key: sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=2815366577.07..2815366578.51 rows=576
 width=27)
                    ->  XN Hash Join DS_BCAST_INNER  (cost=109.98..2815365714.80
 rows=172456 width=27)
                          Hash Cond: ("outer".eventid = "inner".eventid)
                          ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456
 width=14)
                          ->  XN Hash  (cost=87.98..87.98 rows=8798 width=21)
                                ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798
 width=21)
```

Redistribution des données

La sortie EXPLAIN des jointures spécifie également une méthode permettant de déplacer les données autour d'un cluster pour faciliter la jointure. Ce mouvement des données peut être une diffusion ou une redistribution. Dans une diffusion, les valeurs de données d'un côté d'une jointure sont copiées à partir de chaque nœud de calcul dans tous les autres nœuds de calcul, afin que chaque nœud de calcul se retrouve avec une copie complète des données. Dans une redistribution,

les valeurs de données participantes sont envoyés de leur tranche actuelle vers une nouvelle tranche (éventuellement sur un autre nœud). Les données sont généralement redistribuées pour correspondre à la clé de distribution de l'autre table participant à la jointure si cette clé de distribution est l'une des colonnes de jointure. Si aucune des tables ne dispose de clés de distribution sur l'une des tables de jointure, les deux tables sont distribuées ou la table interne est diffusée à chaque nœud.

La sortie EXPLAIN fait également référence aux tables internes et externes. La table interne est analysée d'abord et s'affiche près de bas du plan de requête. La table interne est la table qui fait l'objet d'une recherche de correspondances. Elle est généralement conservée en mémoire, est généralement la table source pour le hachage et, si possible, est la plus petite des deux tables qui sont jointes. La table externe est la source des lignes à mettre en correspondant avec la table interne. Elle est généralement lue à partir du disque. L'optimiseur de requête choisit la table interne et la table externe en fonction des statistiques de la base de données obtenues lors de la dernière exécution de la commande ANALYZE. L'ordre des tables dans la clause FROM d'une requête ne détermine pas quelle table est interne et quelle table est externe.

Utilisez les attributs suivants dans les plans de requête pour identifier la manière dont les données sont déplacées afin de simplifier une requête :

- DS_BCAST_INNER

  Une copie de la totalité de la table interne est diffusée à tous les nœuds de calcul.
- DS_DIST_ALL_NONE

  Aucun redistribution n'est obligatoire, car la table interne a déjà été distribuée à chaque nœud à l'aide de DISTSTYLE ALL.
- DS_DIST_NONE

  Aucune table n'est redistribuée. Les jointures colocalisées sont possibles, car les tranches correspondantes sont jointes sans transfert de données entre les nœuds.
- DS_DIST_INNER

  La table interne est redistribuée.
- DS_DIST_OUTER

  La table externe est redistribuée.
- DS_DIST_ALL_INNER

La totalité de la table interne est redistribué à une seule tranche, car la table externe utilise DISTSTYLE ALL.

- DS_DIST_BOTH

  Les deux tables sont redistribuées.

Révision des étapes du plan de requête

Vous pouvez voir les étapes dans un plan de requête en exécutant la commande EXPLAIN. L'exemple suivant présente une requête SQL et commente la sortie. En lisant le plan de requête en partant du bas, vous pouvez voir chacune des opérations logiques utilisées pour exécuter la requête. Pour de plus amples informations, veuillez consulter Création et interprétation d'un plan de requêtes.

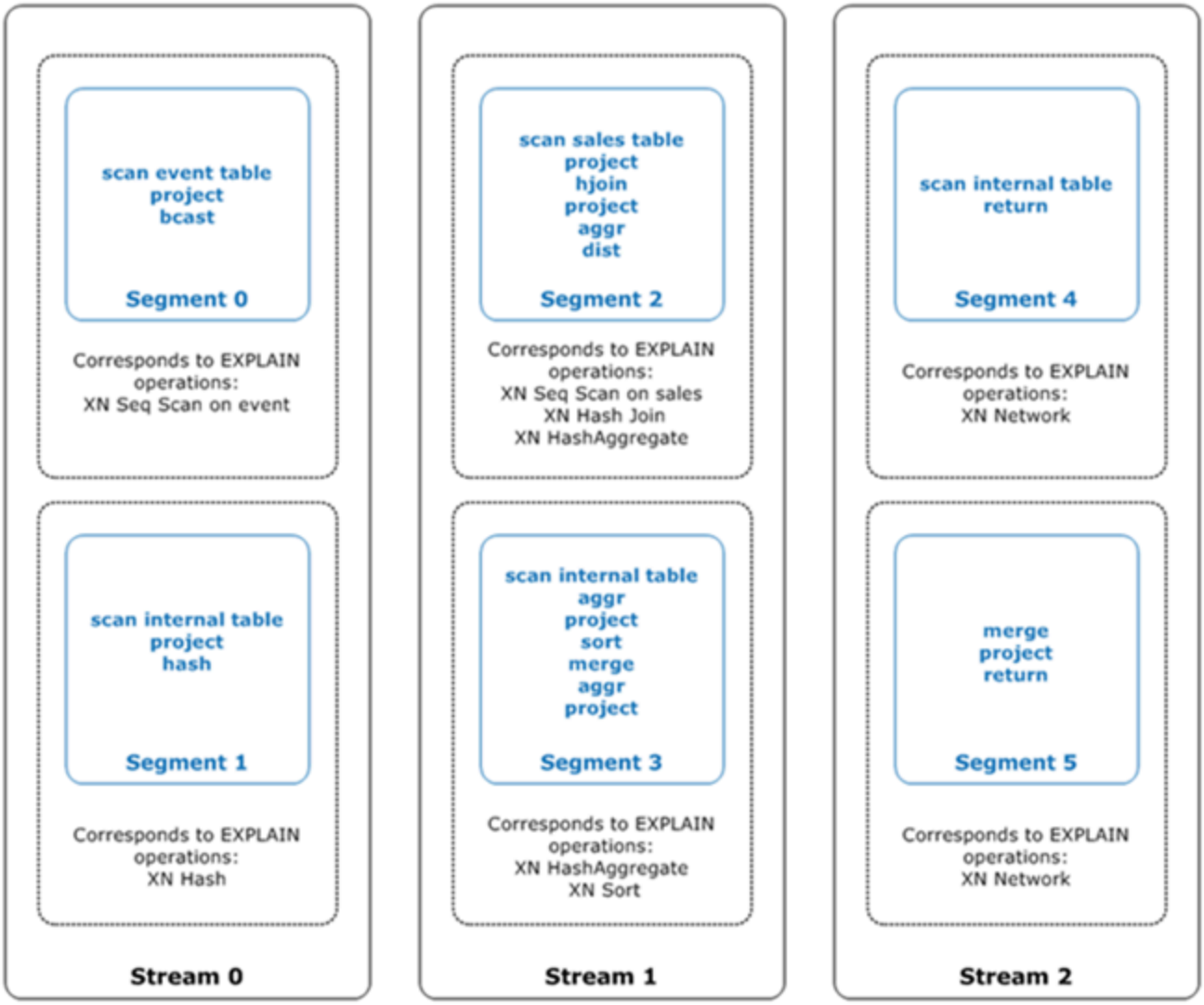
```
explain
select eventname, sum(pricepaid) from sales, event
where sales.eventid = event.eventid
group by eventname
order by 2 desc;
```

```
XN Merge  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
  Merge Key: sum(sales.pricepaid)
  ->  XN Network  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
        Send to leader
        ->  XN Sort  (cost=1002815366604.92..1002815366606.36 rows=576 width=27)
              Sort Key: sum(sales.pricepaid)
              ->  XN HashAggregate  (cost=2815366577.07..2815366578.51 rows=576
 width=27)
                    ->  XN Hash Join DS_BCAST_INNER  (cost=109.98..2815365714.80
 rows=172456 width=27)
                          Hash Cond: ("outer".eventid = "inner".eventid)
                          ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456
 width=14)
                          ->  XN Hash  (cost=87.98..87.98 rows=8798 width=21)
                                ->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798
 width=21)
```

Dans le cadre de la génération d'un plan de requête, l'optimiseur de requêtes décompose le plan en flux, segments et étapes. L'optimiseur de requêtes décompose le plan pour préparer la distribution des données et de la charge de travail de requête entre les nœuds de calcul. Pour plus d'informations

sur les flux, les segments et les étapes, consultez Workflow d'exécution et de planification de requête.

L'illustration suivante présente la requête précédente et le plan de requête associé. Il affiche la manière dont les opérations de requête impliquées sont mappées en étapes utilisées par Amazon Redshift pour générer du code compilé pour les tranches de nœud de calcul. Chaque opération de plan de requête correspond à plusieurs étapes dans les segments et parfois à plusieurs segments dans les flux.



Dans cette illustration, l'optimiseur de requêtes exécute le plan de requête comme suit :

1. Dans `Stream 0`, la requête exécute `Segment 0` avec une opération d'analyse séquentielle pour analyser la table `events`. La requête passe à `Segment 1` avec une opération de hachage pour créer la table de hachage pour la table interne dans la jointure.

2. Dans `Stream 1`, la requête exécute `Segment 2` avec une opération d'analyse séquentielle pour analyser la table `sales`. Elle continue avec `Segment 2` et une jointure de hachage pour joindre des tables où les colonnes de jointure ne sont pas à la fois des clés de distribution et des clés de tri. Elle continue encore avec `Segment 2` et un agrégat de hachage pour agréger les résultats. Ensuite, la requête exécute `Segment 3` avec une opération d'agrégation de hachage pour effectuer des fonctions d'agrégat groupées non triées, ainsi qu'une opération de tri pour évaluer la clause ORDER BY et d'autres opérations de tri.
3. Dans `Stream 2`, la requête exécute une opération de réseau dans `Segment 4` et `Segment 5` pour envoyer des résultats intermédiaires au nœud principal pour un traitement ultérieur.

Le dernier segment d'une requête renvoie les données. Si le jeu de retour est agrégé ou trié, les nœuds de calcul envoient chacun leur morceau du résultat intermédiaire au nœud principal. Le nœud principal fusionne ensuite les données afin que le résultat final puisse être renvoyé au client demandeur.

Pour plus d'informations sur les opérateurs EXPLAID, consultez EXPLAIN.

Facteurs affectant la performance des requêtes

Un certain nombre de facteurs peut affecter les performances des requêtes. Les aspects suivants de vos opérations de données, de cluster et de base de données jouent tous un rôle dans la vitesse de traitement de vos requêtes.

- Nombre de nœuds, de processeurs ou de tranches – Un nœud de calcul est partitionné en tranches. Davantage de nœuds signifie davantage de processeurs et de tranches, ce qui accélère le traitement de vos requêtes grâce à l'exécution simultanée de parties de la requête entre les tranches. Cependant, davantage de nœuds signifie également une dépense supérieure, autrement dit, il vous faudra trouver un équilibre entre les coûts et les performances en fonction de votre système. Pour plus d'informations sur l'architecture de cluster Amazon Redshift, consultez Architecture système de l'entrepôt des données.
- Types de nœuds : un cluster Amazon Redshift peut utiliser l'un des nombreux types de nœuds. Chaque type de nœud propose différentes tailles et limites vous permettant de mettre à l'échelle votre cluster de façon appropriée. La taille de nœud détermine la capacité de stockage, la mémoire, le processeur et le prix de chaque nœud du cluster. Pour plus d'informations sur les types de nœuds, consultez Présentation des clusters Amazon Redshift dans le Guide de gestion Amazon Redshift.

- Distribution des données – Amazon Redshift stocke les données de la table sur les nœuds de calcul en fonction du style de distribution de la table. Lorsque vous exécutez une requête, l'optimiseur de requête redistribue les données sur les nœuds de calcul en fonction des besoins afin d'effectuer des jointures et des agrégations. Le choix du style de distribution adapté à une table permet de réduire l'impact de l'étape de la redistribution en plaçant les données à l'emplacement souhaité avant que les jointures soient effectuées. Pour de plus amples informations, veuillez consulter Distribution des données pour l'optimisation des requêtes.
- Ordre de tri des données – Amazon Redshift stocke les données de la table sur le disque dans un ordre trié selon les clés de tri de la table. L'optimiseur de requête et le processeur de requêtes utilisent les informations sur l'emplacement des données afin de réduire le nombre de blocs à analyser, ce qui améliore ainsi la vitesse de la requête. Pour de plus amples informations, veuillez consulter Clés de tri.
- Taille du jeu de données – Un volume de données plus élevé dans le cluster peut ralentir les performances des requêtes, car davantage de lignes doivent être analysées et redistribuées. Vous pouvez atténuer cet effet en nettoyant et en archivant régulièrement les données et en utilisant un prédicat afin de limiter le jeu de données de la requête.
- Opérations simultanées – L'exécution de plusieurs opérations simultanément peut affecter les performances des requêtes. Chaque opération prend un ou plusieurs emplacements dans une file d'attente de requête disponible et utilise la mémoire qui leur est associée. Si d'autres opérations sont en cours d'exécution, il se peut que les emplacements de file d'attente de requête nécessaires ne soient pas disponibles. Dans ce cas, la requête devra attendre que des emplacements s'ouvrent pour commencer le traitement. Pour plus d'informations sur la création et la configuration des files d'attente de requête, consultez Gestion de la charge de travail.
- Structure de requête – La façon dont votre requête est rédigée influe sur ses performances. Écrivez autant que possible des requêtes à traiter et renvoyez aussi peu de données que nécessaire. Pour de plus amples informations, veuillez consulter Bonnes pratiques Amazon Redshift pour la conception de requêtes.
- Compilation de code – Amazon Redshift génère et compile du code pour chaque plan d'exécution de requête.

  Le code compilé s'exécute plus rapidement car il supprime la surcharge liée à l'utilisation d'un interpréteur. Vous avez toujours un coût de surcharge la première fois que le code est généré et compilé. Par conséquent, la première fois que vous exécutez une requête, ses performances peuvent prêter à confusion. Le coût de surcharge peut être particulièrement significatif lorsque vous exécutez des requêtes ponctuelles. Exécutez la requête une deuxième fois pour déterminer ses performances typiques. Amazon Redshift utilise un service de compilation sans serveur

pour dimensionner les compilations de requêtes au-delà des ressources de calcul d'un cluster Amazon Redshift. Les segments de code compilés sont mis en cache localement sur le cluster et dans un cache pratiquement illimité. Ce cache persiste après le redémarrage du cluster. Les exécutions ultérieures de la même requête sont plus rapides car elles peuvent éviter la phase de compilation.

Le cache n'est pas compatible entre les versions d'Amazon Redshift. Par conséquent, le cache de compilation est vidé et le code est recompilé lorsque les requêtes s'exécutent après une mise à niveau de la version. Si vos requêtes sont strictes SLAs, nous vous recommandons de pré-exécuter des segments de requête qui analysent les données des tables de cluster. Cela permet à Amazon Redshift de mettre en cache les données de la table de base, réduisant ainsi le temps de planification des requêtes après une mise à niveau de la version. En utilisant un service de compilation évolutif, Amazon Redshift peut compiler du code en parallèle pour offrir des performances rapides. L'ampleur de l'accélération de la charge de travail dépend de la complexité et de la simultanéité des requêtes.

Analyse et amélioration des requêtes

La récupération d'informations à partir d'un entrepôt des données Amazon Redshift implique l'exécution de requêtes complexes sur des quantités de données extrêmement importantes, ce qui peut prendre beaucoup de temps à traiter. Pour être sûr de traiter les requêtes le plus rapidement possible, il existe un certain nombre d'outils que vous pouvez utiliser pour identifier les problèmes de performances potentiels.

Rubriques

- Flux de travail d'analyse des requêtes
- Révision des alertes de requêtes
- Analyse du plan de requête
- Analyse du résumé de la requête
- Amélioration des performances des requêtes
- Requêtes de diagnostics pour l'ajustement des requêtes

Flux de travail d'analyse des requêtes

Si une requête est plus longue à traiter que prévu, utilisez les étapes suivantes pour identifier et corriger les problèmes susceptibles d'affecter négativement les performances de la requête. Si vous ne savez pas quelles requêtes de votre système peuvent voir leurs performances ajustées, commencez par exécuter la requête diagnostique dans Identification des requêtes particulièrement indiquées pour un ajustement.

1. Assurez-vous que vos tables sont conçues conformément aux bonnes pratiques. Pour de plus amples informations, veuillez consulter Bonnes pratiques Amazon Redshift pour la conception de tables.
2. Vérifiez si vous pouvez supprimer ou archiver des données superflues dans vos tables. Par exemple, supposons que vos requêtes ciblent toujours les six derniers mois de données, mais que vous avez les 18 derniers mois dans vos tables. Dans ce cas, vous pouvez supprimer ou archiver les données plus anciennes afin de réduire le nombre d'enregistrements devant être analysés et distribués.
3. Exécutez la commande VACUUM sur les tables de la requête pour récupérer de l'espace et trier à nouveau les lignes. L'exécution de VACUUM est utile si la région non triée est volumineuse et que la requête utilise la clé de tri dans une jointure ou dans le prédicat.
4. Exécutez la commande ANALYSE sur les tables de la requête pour vous assurer que les statistiques sont à jour. L'exécution de ANALYZE est utile si l'une des tables de la requête a considérablement changé de taille récemment. Si l'exécution d'une commande ANALYZE complète prend trop de temps, exécutez ANALYZE sur une seule colonne pour réduire le temps de traitement. Cette approche mettra toujours à jour les statistiques de taille de la table, laquelle taille est un facteur important dans la planification de la requête.
5. Assurez-vous que votre requête a été exécutée une fois pour chaque type de client (en fonction du type de protocole de connexion utilisé par le client), afin que la requête soit mise à jour et en cache. Cette approche accélère les exécutions suivantes de la requête. Pour de plus amples informations, veuillez consulter Facteurs affectant la performance des requêtes.
6. Vérifiez la table STL_ALERT_EVENT_LOG afin d'identifier et de corriger les éventuels problèmes avec votre requête. Pour de plus amples informations, veuillez consulter Révision des alertes de requêtes.
7. Exécutez la commande EXPLAIN pour obtenir le plan de requête et l'utiliser afin d'optimiser la requête. Pour de plus amples informations, veuillez consulter Analyse du plan de requête.

8. Utilisez les vues SVL_QUERY_SUMMARY et SVL_QUERY_REPORT pour obtenir des informations récapitulatives et les utiliser afin d'optimiser la requête. Pour de plus amples informations, veuillez consulter Analyse du résumé de la requête.

Parfois, une requête devant s'exécuter rapidement est contrainte d'attendre qu'une autre requête plus longue à s'exécuter se termine. Dans ce cas, vous pouvez ne rien avoir à améliorer dans la requête elle-même, mais vous pouvez améliorer les performances globales du système en créant et en utilisant des files d'attente de requête adaptées à chaque type de requête. Pour connaître les temps d'attente des files d'attente pour vos requêtes, consultez Vérification des temps d'attente des requêtes dans les files d'attente. Pour plus d'informations sur la configuration des files d'attente de requête, consultez Gestion de la charge de travail.

Révision des alertes de requêtes

Pour utiliser la table système STL_ALERT_EVENT_LOG afin d'identifier et de corriger d'éventuels problèmes de performances avec votre requête, procédez comme suit :

1. Exécutez la commande suivante pour déterminer l'ID de votre requête :

```
select query, elapsed, substring
from svl_qlog
order by query
desc limit 5;
```

   Examinez le texte de la requête tronquée dans le champ `substring` pour déterminer quelle valeur de `query` sélectionner. Si vous avez exécuté la requête plusieurs fois, utilisez la valeur de `query` de la ligne avec la valeur de `elapsed` inférieure. Il s'agit de la ligne de la version compilée. Si vous avez exécuté un grand nombre de requêtes, vous pouvez augmenter la valeur utilisée par la clause LIMIT utilisée pour vous assurer que votre requête est incluse.

2. Sélectionnez des lignes dans STL_ALERT_EVENT_LOG pour votre requête :

```
Select * from stl_alert_event_log where query = MyQueryID;
```

userid	query	slice	segment	step	pid	xid	event	solution	event_time
100	32359	4	0	0	8780	71195	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-10 17:40:50
100	32359	5	0	0	8781	71195	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-10 17:40:50
100	109142	4	0	0	8780	302411	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-24 20:32:28
100	109142	5	0	0	8781	302411	Very selective query filter:ratio=rows(2)/r	Review the choice of sort key to enable...	2015-02-24 20:32:28
100	109828	4	1	0	8746	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:27:52
100	109828	5	1	0	8747	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:27:52
100	109829	4	1	0	8760	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:28:01
100	109829	5	1	0	8761	304543	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-24 23:28:01
100	113910	4	1	0	8774	316848	Very selective query filter:ratio=rows(3)/r	Review the choice of sort key to enable...	2015-02-25 17:14:58

3. Evaluez les résultats de votre requête. Utilisez le tableau suivant pour rechercher des solutions possibles aux problèmes que vous avez identifiés.

> **Note**
>
> Toutes les requêtes ne contiennent pas des lignes dans STL_ALERT_EVENT_LOG, seules celles ayant des problèmes identifiés.

Problème	Valeur de l'événement	Valeur de la solution	Solution recommandée
Les statistiques des tables de la requête sont manquantes ou ne sont pas à jour.	Statistiques du planificateur de requête manquantes	Exécuter la commande ANALYZE	Consultez Statistiques de table manquantes ou obsolètes.
Le plan de requête contient une jointure de boucle imbriquée (la jointure la moins optimale).	Jointure de boucle imbriquée dans le plan de requête	Vérifiez les prédicats de jointure pour éviter les produits cartésiens	Consultez Boucle imbriquée.
L'analyse a ignoré un certain nombre de lignes qui sont marquées comme supprimées, mais pas vidées, ou des lignes qui ont été ajoutées mais pas la validées.	Analyse d'un grand nombre de lignes supprimées	Exécutez la commande VACUUM pour récupérer l'espace supprimé	Consultez Lignes fantômes ou non validées.

Problème	Valeur de l'événement	Valeur de la solution	Solution recommandée
Plus de 1 000 000 de lignes ont été redistribuées pour une jointure par hachage ou une agrégation.	Distribution d'un grand nombre de lignes sur le réseau : des RowCount lignes ont été distribuées afin de traiter l'agrégation	Vérifiez le choix de clé de distribution pour colocaliser la jointure ou l'agrégation	Consultez Distribution des données sous-optimales.
Plus de 1 000 000 de lignes ont été diffusées pour une jointure par hachage.	Diffusion d'un grand nombre de lignes sur le réseau	Vérifiez le choix de clé de distribution pour colocaliser la jointure et pensez à utiliser des tables distribuées	Consultez Distribution des données sous-optimales.
Un style de redistribution DS_DIST_ALL_INNER a été indiqué dans le plan de requête, ce qui force une exécution en série, car la totalité de la table interne a été redistribuée sur un seul nœud.	DS_DIST_ALL_INNER pour la jointure par hachage dans le plan de requête	Vérifiez le choix de stratégie de distribution pour distribuer la table interne, plutôt qu'externe	Consultez Distribution des données sous-optimales.

Analyse du plan de requête

Exécutez la commande EXPLAIN pour obtenir un plan de requête.

Avant d'analyser le plan de requête, vous devez savoir comment le lire. Si vous ne savez pas lire un plan de requête, nous vous recommandons de lire Création et interprétation d'un plan de requêtes avant de poursuivre.

Pour analyser les données fournies par le plan de requête, procédez comme suit :

1. Identifiez les étapes ayant le coût le plus élevé. Consacrez-vous à les optimiser lorsque vous effectuez les dernières étapes.
2. Observez les types de jointure :
   - Boucle imbriquée : ces jointures se produisent généralement parce qu'une condition de jointure a été omise. Pour connaître les solutions recommandées, consultez Boucle imbriquée.
   - Hachage et Joindre par hachage : les jointures par hachage sont utilisées pour joindre des tables dans lesquelles les colonnes de jointure ne sont pas des clés de distribution, ni des clés de tri. Pour connaître les solutions recommandées, consultez Joindre par hachage.
   - Joindre par fusion : aucune modification n'est nécessaire.
3. Notez quelle table est utilisée pour la jointure interne et quelle table est utilisée pour la jointure externe. Le moteur de requête choisit généralement la plus petite table pour la jointure interne et la plus grande table pour la jointure externe. Si ce choix ne se fait pas, vos statistiques ne sont vraisemblablement pas à jour. Pour connaître les solutions recommandées, consultez Statistiques de table manquantes ou obsolètes.
4. Vérifiez s'il existe des opérations de tri onéreuses. Si c'est le cas, consultez Lignes non triées ou mal triées pour connaître les solutions recommandées.
5. Recherchez les opérateurs de diffusion suivants là où il y a des opérations onéreuses :
   - DS_BCAST_INNER : indique que la table est diffusée à tous les nœuds de calcul. Ceci convient pour une petite table, mais n'est pas idéal pour une table de taille plus importante.
   - DS_DIST_ALL_INNER : indique que l'ensemble de la charge de travail est placé sur une seule tranche.
   - DS_DIST_BOTH : indique une redistribution intensive.

   Pour connaître les solutions recommandées pour ces situations, consultez Distribution des données sous-optimales.

Analyse du résumé de la requête

Pour obtenir des statistiques et des étapes d'exécution plus détaillées que dans le plan de requête généré par EXPLAIN, utilisez les vues système SVL_QUERY_SUMMARY et SVL_QUERY_REPORT.

SVL_QUERY_SUMMARY fournit des statistiques de requête par flux. Vous pouvez utiliser les informations fournies pour identifier les problèmes posés par les étapes onéreuses, de longue durée et qui écrivent sur le disque.

La vue système SVL_QUERY_REPORT vous permet de consulter des informations similaires à celles de SVL_QUERY_SUMMARY, uniquement par tranche de nœuds de calcul, et non par flux. Vous pouvez utiliser les informations au niveau de la tranche pour identifier la distribution inégale de données dans le cluster (également appelée asymétrie de la distribution de données), qui force certains nœuds à travailler davantage que d'autres et affecte les performances des requêtes.

Rubriques

- Utilisation de la vue SVL_QUERY_SUMMARY
- Utilisation de la vue SVL_QUERY_REPORT
- Mappage du plan de requête au résumé de la requête

Utilisation de la vue SVL_QUERY_SUMMARY

Pour analyser les informations récapitulatives des requêtes par flux en utilisantSVL_QUERY_SUMMARY, procédez comme suit :

1. Exécutez la requête suivante pour déterminer l'ID de votre requête :

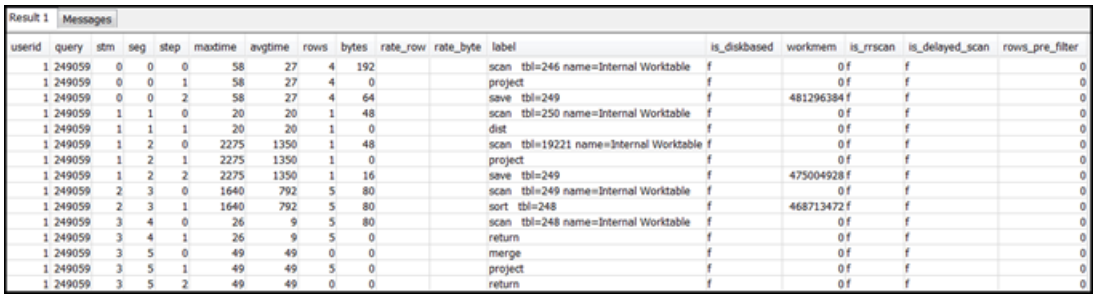
   ```
   select query, elapsed, substring
   from svl_qlog
   order by query
   desc limit 5;
   ```

   Examinez le texte de la requête tronquée dans le champ `substring` pour déterminer quelle valeur de `query` représente votre requête. Si vous avez exécuté la requête plusieurs fois, utilisez la valeur de `query` de la ligne avec la valeur de `elapsed` inférieure. Il s'agit de la ligne de la version compilée. Si vous avez exécuté un grand nombre de requêtes, vous pouvez augmenter la valeur utilisée par la clause LIMIT utilisée pour vous assurer que votre requête est incluse.

2. Sélectionnez les lignes de SVL_QUERY_SUMMARY pour votre requête. Ordonnez les résultats par flux, par segment et par étape :

   ```
   select * from svl_query_summary where query = MyQueryID order by stm, seg, step;
   ```

Voici un exemple de résultat.



Result 1 Messages

userid	query	stm	seg	step	maxtime	avgtime	rows	bytes	rate_row	rate_byte	label	is_diskbased	workmem	is_rrscan	is_delayed_scan	rows_pre_filter
1	249059	0	0	0	58	27	4	192			scan tbl=246 name=Internal Worktable	f	0	f	f	0
1	249059	0	0	1	58	27	4	0			project	f	0	f	f	0
1	249059	0	0	2	58	27	4	64			save tbl=249	f	481296384	f	f	0
1	249059	1	1	0	20	20	1	48			scan tbl=250 name=Internal Worktable	f	0	f	f	0
1	249059	1	1	1	20	20	1	0			dist	f	0	f	f	0
1	249059	1	2	0	2275	1350	1	48			scan tbl=19221 name=Internal Worktable	f	0	f	f	0
1	249059	1	2	1	2275	1350	1	0			project	f	0	f	f	0
1	249059	1	2	2	2275	1350	1	16			save tbl=249	f	475004928	f	f	0
1	249059	2	3	0	1640	792	5	80			scan tbl=249 name=Internal Worktable	f	0	f	f	0
1	249059	2	3	1	1640	792	5	80			sort tbl=248	f	468713472	f	f	0
1	249059	3	4	0	26	9	5	80			scan tbl=248 name=Internal Worktable	f	0	f	f	0
1	249059	3	4	1	26	9	5	0			return	f	0	f	f	0
1	249059	3	5	0	49	49	0	0			merge	f	0	f	f	0
1	249059	3	5	1	49	49	5	0			project	f	0	f	f	0
1	249059	3	5	2	49	49	0	0			return	f	0	f	f	0

3. Mappe les étapes aux opérations du plan de requête à l'aide des informations de Mappage du plan de requête au résumé de la requête. Elles doivent avoir à peu près les mêmes valeurs pour les lignes et les octets (lignes * largeur dans le plan de requête). Si ce n'est pas le cas, consultez Statistiques de table manquantes ou obsolètes pour connaître les solutions recommandées.
4. Vérifiez si le champ `is_diskbased` a une valeur de `t` (true) pour n'importe quelle étape. Les hachages, les agrégats et les tris sont les opérateurs susceptibles d'écrire des données sur le disque si le système ne dispose pas de suffisamment de mémoire allouée pour le traitement des requêtes.

   Si `is_diskbased` a la valeur true, consultez Mémoire insuffisante allouée à la requête pour connaître les solutions recommandées.
5. Passez en revue les valeurs des `label` champs et vérifiez s'il existe une AGG-DIST-AGG séquence quelconque dans les étapes. Sa présence indique une agrégation en deux étapes, ce qui est onéreux. Pour résoudre ce problème, modifiez la clause GROUP BY pour utiliser la clé de distribution (la première clé, s'il y en a plusieurs).
6. Vérifiez la valeur de `maxtime` de chaque segment (identique à toutes les étapes du segment). Identifiez le segment ayant la valeur de `maxtime` la plus élevée et vérifiez les étapes de ce segment pour les opérateurs suivants.

> **Note**
>
> Une valeur de `maxtime` élevée n'indique pas nécessairement de problème avec le segment. Malgré une valeur élevée, le temps de traitement du segment pourrait ne pas avoir été long. Tous les segments d'un flux sont chronométrés ensemble. Toutefois, certains segments en aval ne sont peut-être pas en mesure de s'exécuter tant qu'ils n'obtiennent pas de données de ceux situés en amont. Cela peut donner l'impression qu'ils

ont pris beaucoup de temps, car leur valeur `maxtime` inclura leur temps d'attente et leur temps de traitement.

- BCAST ou DIST : dans ces cas, la valeur de `maxtime` élevée peut être le résultat d'une redistribution d'un grand nombre de lignes. Pour connaître les solutions recommandées, consultez Distribution des données sous-optimales.
- HJOIN (jointure par hachage) : si l'étape en question a une valeur très élevée dans le champ `rows` par rapport à la valeur de `rows` dans l'étape RETURN finale dans la requête, consultez Joindre par hachage pour connaître les solutions recommandées.
- SCAN/SORT : recherchez une séquence d'étapes SCAN, SORT, SCAN, MERGE juste avant une étape de jointure. Ce modèle indique que des données non triées sont analysées, triées, puis fusionnées avec la région triée de la table.

  Vérifiez si la valeur des lignes de l'étape SCAN est très élevée par rapport à la valeur des lignes de l'étape RETURN finale dans la requête. Ce modèle indique que le moteur d'exécution analyse des lignes qui sont ignorées par la suite, ce qui est inefficace. Pour connaître les solutions recommandées, consultez Prédicat pas assez restrictif.

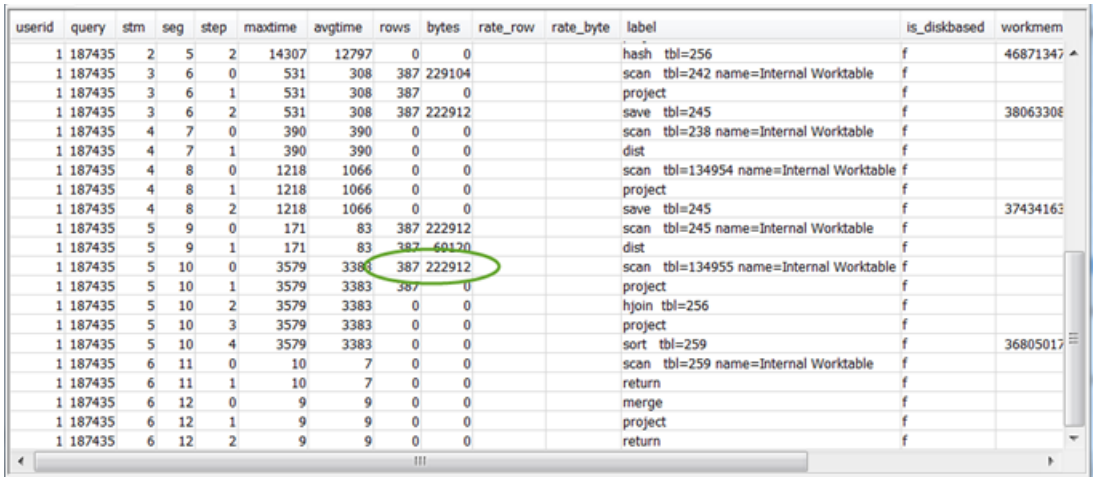
  Si la valeur `maxtime` de l'étape SCAN est élevée, consultez Clause WHERE sous-optimale pour connaître les solutions recommandées.

  Si la valeur `rows` de l'étape SORT n'est pas égale à zéro, consultez Lignes non triées ou mal triées pour connaître les solutions recommandées.

7. Vérifiez les valeurs `rows` et `bytes` des étapes 5 à 10 précédant l'étape RETURN finale pour savoir quelle est la quantité de données renvoyée au client. Ce processus peut être complexe.

   Par exemple, dans l'exemple de résumé de requête suivant, la troisième étape PROJECT fournit une `rows` valeur, mais pas de `bytes` valeur. En parcourant les étapes précédentes pour en trouver une ayant la même `rows` valeur, vous trouverez l'étape SCAN qui fournit à la fois des informations sur les lignes et les octets.

   Voici un exemple de résultat.



userid	query	stm	seg	step	maxtime	avgtime	rows	bytes	rate_row	rate_byte	label	is_diskbased	workmem
1	187435	2	5	2	14307	12797	0	0			hash tbl=256	f	46871347
1	187435	3	6	0	531	308	387	229104			scan tbl=242 name=Internal Worktable	f	
1	187435	3	6	1	531	308	387	0			project	f	
1	187435	3	6	2	531	308	387	222912			save tbl=245	f	38063308
1	187435	4	7	0	390	390	0	0			scan tbl=238 name=Internal Worktable	f	
1	187435	4	7	1	390	390	0	0			dist	f	
1	187435	4	8	0	1218	1066	0	0			scan tbl=134954 name=Internal Worktable	f	
1	187435	4	8	1	1218	1066	0	0			project	f	
1	187435	4	8	2	1218	1066	0	0			save tbl=245	f	37434163
1	187435	5	9	0	171	83	387	222912			scan tbl=245 name=Internal Worktable	f	
1	187435	5	9	1	171	83	387	60120			dist	f	
1	187435	5	10	0	3579	3383	387	222912			scan tbl=134955 name=Internal Worktable	f	
1	187435	5	10	1	3579	3383	387	0			project	f	
1	187435	5	10	2	3579	3383	0	0			hjoin tbl=256	f	
1	187435	5	10	3	3579	3383	0	0			project	f	
1	187435	5	10	4	3579	3383	0	0			sort tbl=259	f	36805017
1	187435	6	11	0	10	7	0	0			scan tbl=259 name=Internal Worktable	f	
1	187435	6	11	1	10	7	0	0			return	f	
1	187435	6	12	0	9	9	0	0			merge	f	
1	187435	6	12	1	9	9	0	0			project	f	
1	187435	6	12	2	9	9	0	0			return	f	

Si vous renvoyez un volume inhabituellement élevé de données, consultez Ensemble de résultats très volumineux pour connaître les solutions recommandées.

8. Vérifiez si la valeur de `bytes` est élevée par rapport à la valeur de `rows` pour n'importe quelle étape, par rapport aux autres étapes. Ce modèle peut indiquer que vous sélectionnez un grand nombre de colonnes. Pour connaître les solutions recommandées, consultez Longue liste SELECT.

Utilisation de la vue SVL_QUERY_REPORT

Pour analyser les informations récapitulatives des requêtes par tranche à l'aide deSVL_QUERY_REPORT, procédez comme suit :

1. Exécutez la commande suivante pour déterminer l'ID de votre requête :

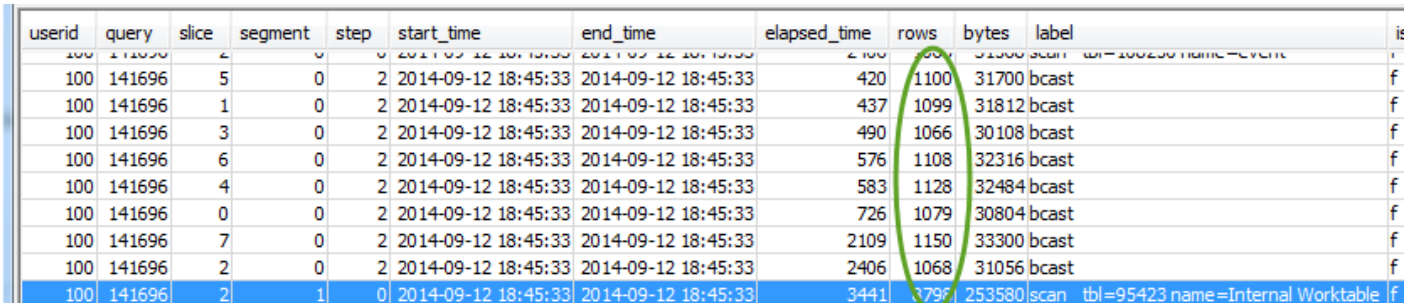
```
select query, elapsed, substring
from svl_qlog
order by query
desc limit 5;
```

Examinez le texte de la requête tronquée dans le champ `substring` pour déterminer quelle valeur de `query` représente votre requête. Si vous avez exécuté la requête plusieurs fois, utilisez la valeur de `query` de la ligne avec la valeur de `elapsed` inférieure. Il s'agit de la ligne de la version compilée. Si vous avez exécuté un grand nombre de requêtes, vous pouvez augmenter la valeur utilisée par la clause LIMIT utilisée pour vous assurer que votre requête est incluse.

2. Sélectionner des lignes dans SVL_QUERY_REPORT pour votre requête. Ordonnez les résultats par segment, par étape, par elapsed_time et par lignes :

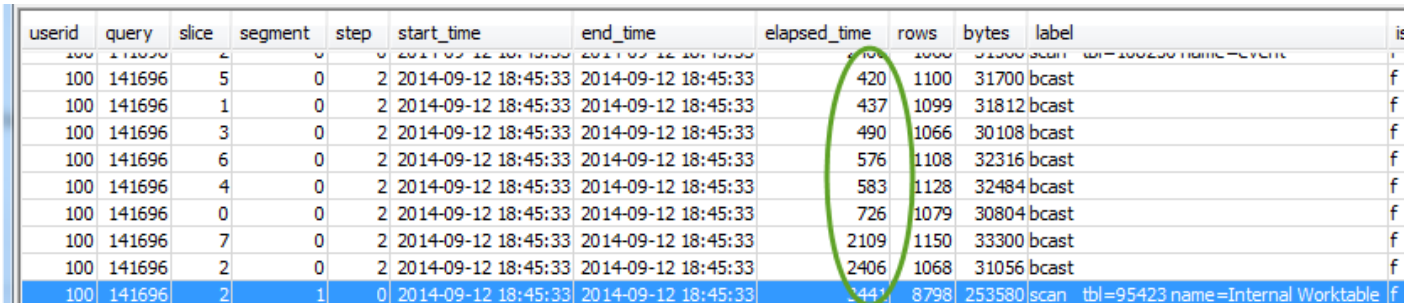
```
select * from svl_query_report where query = MyQueryID order by segment, step,
 elapsed_time, rows;
```

3. Pour chaque étape, vérifiez que toutes les tranches traitent à peut près le même nombre de lignes :



userid	query	slice	segment	step	start_time	end_time	elapsed_time	rows	bytes	label	is
100	141696	5	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	420	1100	31700	bcast	f
100	141696	1	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	437	1099	31812	bcast	f
100	141696	3	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	490	1066	30108	bcast	f
100	141696	6	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	576	1108	32316	bcast	f
100	141696	4	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	583	1128	32484	bcast	f
100	141696	0	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	726	1079	30804	bcast	f
100	141696	7	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2109	1150	33300	bcast	f
100	141696	2	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2406	1068	31056	bcast	f
100	141696	2	1	0	2014-09-12 18:45:33	2014-09-12 18:45:33	3441	8798	253580	scan tbl=95423 name=Internal Worktable	f

Vérifiez également que toutes les tranches prennent à peu près autant de temps :



userid	query	slice	segment	step	start_time	end_time	elapsed_time	rows	bytes	label	is
100	141696	5	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	420	1100	31700	bcast	f
100	141696	1	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	437	1099	31812	bcast	f
100	141696	3	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	490	1066	30108	bcast	f
100	141696	6	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	576	1108	32316	bcast	f
100	141696	4	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	583	1128	32484	bcast	f
100	141696	0	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	726	1079	30804	bcast	f
100	141696	7	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2109	1150	33300	bcast	f
100	141696	2	0	2	2014-09-12 18:45:33	2014-09-12 18:45:33	2406	1068	31056	bcast	f
100	141696	2	1	0	2014-09-12 18:45:33	2014-09-12 18:45:33	3441	8798	253580	scan tbl=95423 name=Internal Worktable	f

Si ces valeurs sont très différences, cela peut révéler une asymétrie de la distribution des données due à un style de distribution sous-optimal pour cette requête particulière. Pour connaître les solutions recommandées, consultez Distribution des données sous-optimales.

Mappage du plan de requête au résumé de la requête

Lorsque vous analysez le résumé de la requête, vous pouvez obtenir plus de détails en mappant les opérations du plan de requête aux étapes (identifiées par les valeurs des champs d'étiquette) dans le résumé de la requête. Le tableau suivant fait correspondre les opérations du plan de requête aux étapes récapitulatives des requêtes.

Opération de plan de requête	Valeur du champ d'étiquette	Description
Regrouper HashAggregate GroupAggregate	AGGR	Evalue les fonctions d'agrégation et les conditions GROUP BY.
DS_BCAST_INNER	BCAST (broadcast)	Diffuse une table entière ou un ensemble de lignes (par exemple, un ensemble filtré de lignes d'une table) à tous les nœuds.
N'apparaît pas dans le plan de requête	DELETE	Supprime les lignes des tables.
DS_DIST_NONE DS_DIST_ALL_NONE DS_DIST_INNER DS_DIST_ALL_INNER DS_DIST_ALL_BOTH	DIST (distribute)	Distribue des lignes aux nœuds à des fins de jointure parallèle ou d'autres traitements parallèles.
HASH	HASH	Créer une table de hachage à utiliser dans les jointures par hachage.
Joindre par hachage	HJOIN (hash join)	Effectue une jointure par hachage de deux tables ou de jeux de résultats intermédiaires.
N'apparaît pas dans le plan de requête	INSERT	Insère des lignes dans les tables.

Opération de plan de requête	Valeur du champ d'étiquette	Description
Limite	LIMIT	Applique une clause LIMIT à des jeux de résultats.
Fusionner	MERGE	Fusionne les lignes dérivées d'opérations de tri ou de jointure parallèles.
Joindre par fusion	MJOIN (merge join)	Effectue une jointure par fusion de deux tables ou de jeux de résultats intermédi aires.
Boucle imbriquée	NLOOP (nested loop)	Effectue une jointure de boucle imbriquée de deux tables ou de jeux de résultats intermédiaires.
N'apparaît pas dans le plan de requête	PARSE	Analyse des chaînes en valeurs binaires pour le chargement.
Projet	PROJECT	Evalue les expressions.
Réseau	RETURN	Renvoie des lignes au principal ou au client.
N'apparaît pas dans le plan de requête	SAVE	Matérialise des lignes à utiliser à la prochaine étape du traitement.
Seq Scan	SCAN	Analyse des tables ou des jeux de résultats intermédi aires.

Opération de plan de requête	Valeur du champ d'étiquette	Description
Tri	SORT	Trie des lignes ou des jeux de résultat intermédiaires définis comme obligatoires par d'autres opérations ultérieures (comme des jointures ou des agrégations) ou pour satisfaire une clause ORDER BY.
Unique	UNIQUE	Applique une clause SELECT DISTINCT ou supprimer des doublons si d'autres opération s l'exigent.
Fenêtre	WINDOW	Calcule des fonctions de fenêtrage d'agrégation ou de classement.

Amélioration des performances des requêtes

Vous trouverez ci-dessous quelques problèmes courants qui affectent les performances des requêtes Amazon Redshift, ainsi que des instructions sur les moyens de les diagnostiquer et de les résoudre.

Rubriques

- Statistiques de table manquantes ou obsolètes
- Boucle imbriquée
- Joindre par hachage
- Lignes fantômes ou non validées
- Lignes non triées ou mal triées
- Distribution des données sous-optimales
- Mémoire insuffisante allouée à la requête
- Clause WHERE sous-optimale
- Prédicat pas assez restrictif
- Ensemble de résultats très volumineux

- Longue liste SELECT

Statistiques de table manquantes ou obsolètes

Si des statistiques de la table sont manquantes ou obsolètes, ce qui suit peut s'afficher :

- Un message d'avertissement dans les résultats de la commande EXPLAIN.
- Un événement d'alerte de statistiques manquantes dans STL_ALERT_EVENT_LOG. Pour de plus amples informations, veuillez consulter Révision des alertes de requêtes.

Pour résoudre ce problème, exécutez ANALYSE.

Boucle imbriquée

Si une boucle imbriquée est présente, un événement d'alerte de boucle imbriquée peut s'afficher dans STL_ALERT_EVENT_LOG. Vous pouvez également identifier ce type d'événement en exécutant la requête de la section Identification des requêtes avec des boucles imbriquées. Pour de plus amples informations, veuillez consulter Révision des alertes de requêtes.

Pour résoudre ce problème, vérifiez s'il existe des jointures croisées dans votre requête et supprimez-les si possible. Les jointures croisées sont des jointures sans condition de jointure qui entraînent le produit cartésien de deux tables. Elles sont généralement exécutées en tant que jointures de boucle imbriquée, qui sont les types de jointures les plus lents possibles.

Joindre par hachage

Si une jointure par hachage est présente, les éléments suivants peuvent s'afficher :

- Des opérations de hachage et de jointure par hachage dans le plan de la requête. Pour de plus amples informations, veuillez consulter Analyse du plan de requête.
- Une étape HJOIN dans le segment avec la valeur maxtime la plus élevée dans SVL_QUERY_SUMMARY. Pour de plus amples informations, veuillez consulter Utilisation de la vue SVL_QUERY_SUMMARY.

Pour résoudre ce problème, vous pouvez adopter deux approches :

- Réécrivez la requête afin d'utiliser une jointure par fusion si possible. Pour cela, spécifiez les colonnes de jointure qui sont des clés de distribution et des clés de tri.

- Si l'étape HJOIN de SVL_QUERY_SUMMARY a une très grande valeur dans le champ des lignes par rapport à la valeur des lignes de l'étape RETURN finale de la requête, vérifiez si vous pouvez réécrire la requête afin d'effectuer une jointure avec une colonne unique. Lorsqu'une requête n'effectue pas de jointure avec une colonne unique, telle qu'une clé primaire, cela augmente le nombre de lignes impliquées dans la jointure.

Lignes fantômes ou non validées

Si des lignes fantômes ou non validées sont présentes, un événement d'alerte peut s'afficher dans STL_ALERT_EVENT_LOG indiquant un nombre de lignes fantômes excessif. Pour de plus amples informations, veuillez consulter Révision des alertes de requêtes.

Pour résoudre ce problème, vous pouvez adopter deux approches :

- Vérifiez l'onglet Charges de votre console Amazon Redshift pour les opérations de charge actives sur l'une des tables de requête. Si vous voyez des opérations de chargement actives, attendez qu'elles se terminent avant d'agir.
- S'il n'y a aucune opération de charge active, exécutez VACUUM sur les tables de la requête pour supprimer les lignes supprimées.

Lignes non triées ou mal triées

Si des lignes non triées ou mal triées sont présentes, un événement d'alerte de filtre très sélectif peut s'afficher dans STL_ALERT_EVENT_LOG. Pour de plus amples informations, veuillez consulter Révision des alertes de requêtes.

Vous pouvez également vérifier si l'une des tables de votre requête présente de grandes zones non triés en exécutant la requête Identification des tables comportant une asymétrie des données ou des lignes non triées.

Pour résoudre ce problème, vous pouvez adopter deux approches :

- Exécutez VACUUM sur les tables de requête pour trier à nouveau les lignes.
- Vérifiez les clés de tri des tables de la requête pour voir si des améliorations peuvent être apportées. N'oubliez pas de comparer les performances de cette requête par rapport aux performances d'autres requêtes importantes et au système global avant toute modification. Pour de plus amples informations, veuillez consulter Clés de tri.

Distribution des données sous-optimales

Si la distribution des données est sous-optimale, les éléments suivants peuvent s'afficher :

- Un événement d'alerte d'exécution en série, de grande diffusion ou de grande distribution s'affiche dans STL_ALERT_EVENT_LOG. Pour de plus amples informations, veuillez consulter Révision des alertes de requêtes.
- Les tranches ne traitent pas le même nombre de lignes (approximativement) pour une étape donnée. Pour de plus amples informations, veuillez consulter Utilisation de la vue SVL_QUERY_REPORT.
- Les tranches ne prennent pas autant de temps (approximativement) pour une étape donnée. Pour de plus amples informations, veuillez consulter Utilisation de la vue SVL_QUERY_REPORT.

Si rien de ce qui précède ne s'applique, vous pouvez également vérifier si l'une des tables de votre requête comporte une asymétrie des données en exécutant la requête dans Identification des tables comportant une asymétrie des données ou des lignes non triées.

Pour résoudre ce problème, vérifiez les styles de distribution des tables de la requête afin de voir si des améliorations peuvent être apportées. N'oubliez pas de comparer les performances de cette requête par rapport aux performances d'autres requêtes importantes et au système global avant toute modification. Pour de plus amples informations, veuillez consulter Distribution des données pour l'optimisation des requêtes.

Mémoire insuffisante allouée à la requête

Si la mémoire allouée à votre requête est insuffisante, vous pouvez voir qu'une étape de SVL_QUERY_SUMMARY comporte une valeur true pour `is_diskbased`. Pour de plus amples informations, veuillez consulter Utilisation de la vue SVL_QUERY_SUMMARY.

Pour résoudre ce problème, allouez davantage de mémoire à la requête en augmentant temporairement le nombre d'emplacements de requête qu'elle utilise. La gestion de la charge de travail (WLM) réserve des emplacements dans une file d'attente de requête équivalents à au niveau de simultanéité défini pour la file d'attente. Par exemple, une file d'attente avec un niveau de simultanéité de 5 dispose de 5 emplacements. La mémoire affectée à la file d'attente est allouée à part égale à chaque emplacement. L'affectation de plusieurs emplacements à une requête donne à celle-ci accès à la mémoire de tous ces emplacements. Pour plus d'informations sur la procédure permettant d'augmenter temporairement les emplacements d'une requête, consultez wlm_query_slot_count.

Clause WHERE sous-optimale

Si votre clause WHERE entraîne des analyses de tables excessives, une étape SCAN peut s'afficher dans le segment ayant la valeur de `maxtime` la plus élevée dans SVL_QUERY_SUMMARY. Pour de plus amples informations, veuillez consulter Utilisation de la vue SVL_QUERY_SUMMARY.

Pour résoudre ce problème, ajoutez une clause WHERE à la requête basée sur la colonne de tri primaire de la table la plus volumineuse. Cette approche permet de réduire le temps d'analyse. Pour de plus amples informations, veuillez consulter Bonnes pratiques Amazon Redshift pour la conception de tables.

Prédicat pas assez restrictif

Si votre requête a un prédicat qui n'est pas suffisamment restrictif, une étape SCAN peut s'afficher dans le segment avec la valeur de `maxtime` la plus élevée dans SVL_QUERY_SUMMARY avec une valeur de `rows` très importante par rapport à la valeur de `rows` de l'étape RETURN finale de la requête. Pour de plus amples informations, veuillez consulter Utilisation de la vue SVL_QUERY_SUMMARY.

Pour résoudre ce problème, essayez d'ajouter un prédicat à la requête ou de rendre le prédicat existant plus restrictifs pour affiner les données de sortie.

Ensemble de résultats très volumineux

Si votre requête renvoie un ensemble de résultats très volumineux, envisagez de réécrire la requête pour utiliser UNLOAD afin d'écrire les résultats sur Amazon S3. Cette approche permettra d'améliorer les performances de l'étape RETURN en tirant parti du traitement parallèle. Pour plus d'informations sur la recherche d'un ensemble de résultats très volumineux, consultez Utilisation de la vue SVL_QUERY_SUMMARY.

Longue liste SELECT

Si votre requête comporte une liste SELECT inhabituellement longue, une valeur de `bytes` élevée peut s'afficher par rapport à la valeur de `rows` de n'importe quelle étape (comparée à d'autres étapes) dans SVL_QUERY_SUMMARY. Cette valeur de `bytes` élevée peut indiquer que vous sélectionnez un grand nombre de colonnes. Pour de plus amples informations, veuillez consulter Utilisation de la vue SVL_QUERY_SUMMARY.

Pour résoudre ce problème, vérifiez les colonnes que vous sélectionnez pour voir si certaines peuvent être supprimées.

Requêtes de diagnostics pour l'ajustement des requêtes

Utilisez les requêtes suivantes pour identifier les problèmes rencontrés par les requêtes ou les tables sous-jacentes susceptibles d'affecter les performances des requêtes. Nous recommandons d'utiliser ces requêtes conjointement aux processus d'ajustement des requêtes expliqués dans la section Analyse et amélioration des requêtes.

> **Note**
>
> Ces requêtes concernent les clusters provisionnés par Amazon Redshift. Ces requêtes ne sont pas destinées à être utilisées avec les groupes de travail Redshift Serverless.

Rubriques

- Identification des requêtes particulièrement indiquées pour un ajustement
- Identification des tables comportant une asymétrie des données ou des lignes non triées
- Identification des requêtes avec des boucles imbriquées
- Vérification des temps d'attente des requêtes dans les files d'attente
- Vérification des alertes de requêtes par table
- Identification des tables avec des statistiques manquantes

Identification des requêtes particulièrement indiquées pour un ajustement

La requête suivante identifie les 50 instructions les plus chronophages exécutées au cours des 7 derniers jours. Vous pouvez utiliser les résultats pour identifier les requêtes inhabituellement longues. Vous pouvez également identifier les requêtes exécutées fréquemment (celles qui apparaissent plusieurs fois dans l'ensemble de résultats). Ces requêtes nécessitent souvent un ajustement en vue d'améliorer les performances du système.

Cette requête fournit également un nombre d'événements d'alertes associés à chaque requête identifiée. Ces alertes fournissent des détails utiles pour améliorer les performances de la requête. Pour de plus amples informations, veuillez consulter Révision des alertes de requêtes.

```
select trim(database) as db, count(query) as n_qry,
max(substring (qrytext,1,80)) as qrytext,
min(run_minutes) as "min" ,
max(run_minutes) as "max",
```

```
avg(run_minutes) as "avg", sum(run_minutes) as total,
max(query) as max_query_id,
max(starttime)::date as last_run,
sum(alerts) as alerts, aborted
from (select userid, label, stl_query.query,
trim(database) as database,
trim(querytxt) as qrytext,
md5(trim(querytxt)) as qry_md5,
starttime, endtime,
(datediff(seconds, starttime,endtime)::numeric(12,2))/60 as run_minutes,
alrt.num_events as alerts, aborted
from stl_query
left outer join
(select query, 1 as num_events from stl_alert_event_log group by query ) as alrt
on alrt.query = stl_query.query
where userid <> 1 and starttime >=  dateadd(day, -7, current_date))
group by database, label, qry_md5, aborted
order by total desc limit 50;
```

Identification des tables comportant une asymétrie des données ou des lignes non triées

La requête suivante identifie les tables comportant une distribution irrégulière de données (asymétrie des données) ou un pourcentage élevé de lignes non triées.

Un valeur de `skew` faible indique que les données de la table sont correctement distribuées. Si une table comporte une valeur de `skew` de 4,00 ou plus, envisagez de modifier le style de distribution de données. Pour de plus amples informations, veuillez consulter Distribution des données sous-optimales.

Si une table comporte une valeur de `pct_unsorted` supérieure à 20 pour cent, envisagez d'exécuter la commande VACUUM. Pour de plus amples informations, veuillez consulter Lignes non triées ou mal triées.

Vous devez également examiner les valeurs de `mbytes` et de `pct_of_total` pour chaque table. Ces colonnes identifient la taille de la table et le pourcentage d'espace sur le disque brut que la table consomme. L'espace disque brut comprend l'espace qui est réservé par Amazon Redshift pour un usage interne, il est donc plus grand que la capacité nominale du disque, qui est la quantité d'espace disque disponible pour l'utilisateur. Ces informations vous permettront de vérifier que l'espace disque est égal à au moins 2,5 fois la taille de votre table la plus volumineuse. Cet espace disponible

permet au système d'écrire des résultats intermédiaires sur le disque lors du traitement de requêtes complexes.

```
select trim(pgn.nspname) as schema,
trim(a.name) as table, id as tableid,
decode(pgc.reldiststyle,0, 'even',1,det.distkey ,8,'all') as distkey,
 dist_ratio.ratio::decimal(10,4) as skew,
det.head_sort as "sortkey",
det.n_sortkeys as "#sks", b.mbytes,
decode(b.mbytes,0,0,((b.mbytes/part.total::decimal)*100)::decimal(5,2)) as
 pct_of_total,
decode(det.max_enc,0,'n','y') as enc, a.rows,
decode( det.n_sortkeys, 0, null, a.unsorted_rows ) as unsorted_rows ,
decode( det.n_sortkeys, 0, null, decode( a.rows,0,0, (a.unsorted_rows::decimal(32)/
a.rows)*100) )::decimal(5,2) as pct_unsorted
from (select db_id, id, name, sum(rows) as rows,
sum(rows)-sum(sorted_rows) as unsorted_rows
from stv_tbl_perm a
group by db_id, id, name) as a
join pg_class as pgc on pgc.oid = a.id
join pg_namespace as pgn on pgn.oid = pgc.relnamespace
left outer join (select tbl, count(*) as mbytes
from stv_blocklist group by tbl) b on a.id=b.tbl
inner join (select attrelid,
min(case attisdistkey when 't' then attname else null end) as "distkey",
min(case attsortkeyord when 1 then attname  else null end ) as head_sort ,
max(attsortkeyord) as n_sortkeys,
max(attencodingtype) as max_enc
from pg_attribute group by 1) as det
on det.attrelid = a.id
inner join ( select tbl, max(mbytes)::decimal(32)/min(mbytes) as ratio
from (select tbl, trim(name) as name, slice, count(*) as mbytes
from svv_diskusage group by tbl, name, slice )
group by tbl, name ) as dist_ratio on a.id = dist_ratio.tbl
join ( select sum(capacity) as  total
from stv_partitions where part_begin=0 ) as part on 1=1
where mbytes is not null
order by  mbytes desc;
```

Identification des requêtes avec des boucles imbriquées

La requête suivante identifie les requêtes ayant des événements d'alertes consignés pour des boucles imbriquées. Pour plus d'informations sur la façon de corriger la condition de boucle imbriquée, consultez Boucle imbriquée.

```
select query, trim(querytxt) as SQL, starttime
from stl_query
where query in (
select distinct query
from stl_alert_event_log
where event like 'Nested Loop Join in the query plan%')
order by starttime desc;
```

Vérification des temps d'attente des requêtes dans les files d'attente

La requête suivante affiche combien de temps les requêtes récentes ont attendu avant qu'un emplacement s'ouvre dans une file d'attente avant leur exécution. Si vous voyez une tendance de temps d'attente élevés, vous pouvez modifier la configuration de votre file d'attente de requête pour obtenir un débit plus élevé. Pour de plus amples informations, veuillez consulter Implémentation de la gestion manuelle de la charge de travail.

```
select trim(database) as DB , w.query,
substring(q.querytxt, 1, 100) as querytxt,  w.queue_start_time,
w.service_class as class, w.slot_count as slots,
w.total_queue_time/1000000 as queue_seconds,
w.total_exec_time/1000000 exec_seconds, (w.total_queue_time+w.total_Exec_time)/1000000
 as total_seconds
from stl_wlm_query w
left join stl_query q on q.query = w.query and q.userid = w.userid
where w.queue_start_Time >= dateadd(day, -7, current_Date)
and w.total_queue_Time > 0  and w.userid >1
and q.starttime >= dateadd(day, -7, current_Date)
order by w.total_queue_time desc, w.queue_start_time desc limit 35;
```

Vérification des alertes de requêtes par table

La requête suivante identifie les tables pour lesquelles des événements d'alertes de requêtes ont été consignés, ainsi que les types d'alertes rencontrés le plus souvent.

Si la valeur de `minutes` pour une ligne avec une table identifiée est élevée, vérifiez cette table pour savoir si elle nécessite des opérations de maintenance habituelles, comme l'exécution de ANALYSE ou de VACUUM.

Si la valeur de `count` est élevée pour une ligne, mais que la valeur de `table` est null, exécutez une requête sur STL_ALERT_EVENT_LOG pour la valeur de `event` associée afin d'étudier pourquoi cette alerte est déclenchée si souvent.

```
select trim(s.perm_table_name) as table,
(sum(abs(datediff(seconds, s.starttime, s.endtime)))/60)::numeric(24,0) as minutes,
 trim(split_part(l.event,':',1)) as event,  trim(l.solution) as solution,
max(l.query) as sample_query, count(*)
from stl_alert_event_log as l
left join stl_scan as s on s.query = l.query and s.slice = l.slice
and s.segment = l.segment and s.step = l.step
where l.event_time >=  dateadd(day, -7, current_Date)
group by 1,3,4
order by 2 desc,6 desc;
```

Identification des tables avec des statistiques manquantes

La requête suivante fournit un nombre de requêtes que vous exécutez sur les tables pour lesquelles des statistiques sont manquantes. Si cette requête renvoie toutes les lignes, examinez la valeur de `plannode` afin de déterminer la table concernée et exécutez ANALYSE sur celle-ci.

```
select substring(trim(plannode),1,100) as plannode, count(*)
from stl_explain
where plannode like '%missing statistics%'
group by plannode
order by 2 desc;
```

Résolution des problèmes de requêtes

Cette section fournit une référence rapide pour identifier et résoudre certains des problèmes les plus courants et les plus graves que vous êtes susceptibles de rencontrer avec les requêtes Amazon Redshift.

Rubriques

- Échecs des connexions

- La requête se bloque
- La requête est trop longue
- La charge échoue
- La charge est trop longue
- Le chargement des données est incorrect
- Définition du paramètre de taille d’extraction JDBC

Ces suggestions vous donnent un point de départ pour le dépannage. Pour plus d’informations, vous pouvez également consulter les ressources suivantes.

Pour plus d'informations sur les changements de comportement dans les fonctionnalités Amazon Redshift susceptibles d'affecter votre application, consultez la section Changements de comportement.

- Accès aux clusters et aux bases de données Amazon Redshift
- Optimisation automatique des tables
- Chargement de données dans Amazon Redshift
- Didacticiel : chargement des données à partir d’Amazon S3

Échecs des connexions

Votre connexion de requête peut échouer pour les raisons suivantes. Nous vous suggérons les approches de résolution des problèmes suivantes.

Le client ne parvient pas à se connecter au serveur

Si vous utilisez des certificats SSL ou serveur, commencez par supprimer cette complexité pendant que vous résolvez le problème de connexion. Ensuite, ajoutez à nouveau les certificats SSL ou serveur lorsque vous avez trouvé une solution. Pour plus d’informations, consultez Configurer les options de sécurité pour les connexions dans le Guide de gestion Amazon Redshift.

La connexion est refusée

En général, lorsque vous recevez un message d'erreur indiquant qu'il est impossible d'établir une connexion, cela signifie qu'il y a un problème d'autorisation d'accès au cluster. Pour plus d'informations, consultez La connexion est refusée ou échoue dans le Guide de gestion Amazon Redshift.

La requête se bloque

Votre requête peut être bloquée ou ne plus répondre pour les raisons suivantes. Nous vous suggérons les approches de résolution des problèmes suivantes.

La connexion à la base de données est abandonnée

Réduisez la taille de l'unité de transmission maximale (MTU). La taille de la MTU détermine la taille maximale, en octets, d'un paquet pouvant être transféré dans une trame Ethernet sur votre connexion réseau. Pour plus d'informations, consultez La connexion à la base de données est abandonnée dans le Guide de gestion Amazon Redshift.

La connexion à la base de données arrive à expiration

La connexion de votre client à la base de données semble se bloquer ou arriver à expiration lorsque vous exécutez de longues requêtes, par exemple une commande COPY. Dans ce cas, vous pouvez constater que la console Amazon Redshift affiche que la requête est terminée, mais que l'outil client lui-même semble toujours exécuter la requête. Les résultats de la requête peuvent être manquants ou incomplets en fonction selon le moment où la connexion s'est arrêtée. Cela se produit lorsque les connexions inactives sont arrêtées par un composant réseau intermédiaire. Pour plus d'informations, consultez Problème de délai d'expiration du pare-feu dans le Guide de gestion Amazon Redshift.

out-of-memoryUne erreur côté client se produit avec ODBC

Si votre application cliente utilise une connexion ODBC et que votre requête crée un ensemble de résultats trop volumineux pour contenir en mémoire, vous pouvez diffuser l'ensemble de résultats sur votre application cliente à l'aide d'un curseur. Pour plus d'informations, consultez DECLARE et Considérations relatives aux performances lors de l'utilisation de curseurs.

out-of-memoryUne erreur côté client se produit avec JDBC

Lorsque vous tentez de récupérer de grands ensembles de résultats via une connexion JDBC, vous pouvez rencontrer des erreurs côté client out-of-memory. Pour de plus amples informations, veuillez consulter Définition du paramètre de taille d'extraction JDBC.

Il existe un blocage potentiel

En cas de blocage potentiel, essayez ce qui suit :

- Affichez les tables système STV_LOCKS et STL_TR_CONFLICT pour rechercher les conflits impliquant des mises à jour de plusieurs tables.
- Utilisez la fonction PG_CANCEL_BACKEND pour annuler une ou plusieurs requêtes conflictuelles.
- Utilisez la fonction PG_TERMINATE_BACKEND pour mettre fin à une session, ce qui oblige toutes les transactions en cours d'exécution dans la session terminée à libérer tous les verrous et à restaurer la transaction.
- Planifiez les opérations d'écriture simultanées avec soin. Pour de plus amples informations, veuillez consulter Gestion des opérations d'écriture simultanées.

La requête est trop longue

Votre requête peut prendre trop de temps pour les raisons suivantes. Nous vous suggérons les approches de résolution des problèmes suivantes.

Les tables ne sont pas optimisées

Définissez la clé de tri, le style de distribution et l'encodage de compression des tables afin de tirer pleinement parti du traitement parallèle. Pour de plus amples informations, consultez Optimisation automatique des tables

La requête écrit sur le disque

Vos requêtes peuvent être en train d'écrire sur le disque pendant une partie de l'exécution des requêtes au moins. Pour de plus amples informations, veuillez consulter Amélioration des performances des requêtes.

La requête doit attendre que d'autres requêtes se terminent

Vous pouvez améliorer les performances globales du système en créant des files d'attente de requêtes et en affectant différents types de requêtes aux files d'attente correspondantes. Pour de plus amples informations, veuillez consulter Gestion de la charge de travail.

Les requêtes ne sont pas optimisées

Analysez le plan EXPLAIN pour rechercher des opportunités de réécriture des requêtes ou d'optimisation de la base de données. Pour de plus amples informations, veuillez consulter Création et interprétation d'un plan de requêtes.

La requête nécessite davantage de mémoire pour s'exécuter

Si une requête spécifique nécessite davantage de mémoire, vous pouvez augmenter la mémoire disponible en augmentant le wlm_query_slot_count.

La base de données nécessite une commande VACUUM pour s'exécuter

Exécutez la commande VACUUM chaque fois que vous ajoutez, supprimez ou modifiez un grand nombre de lignes, sauf si vous chargez vos données dans l'ordre de la clé de tri. La commande VACUUM réorganise vos données afin de conserver l'ordre de tri et de restaurer les performances. Pour de plus amples informations, veuillez consulter Exécution de l'opération VACUUM sur les tables.

Ressources supplémentaires pour la résolution des problèmes liés aux requêtes de longue durée

Vous trouverez ci-dessous des rubriques relatives aux vues système et d'autres sections de documentation utiles pour l'ajustement de requêtes :

- La vue système STV_INFLIGHT affiche les requêtes qui s'exécutent actuellement sur le cluster. Il peut être utile de l'utiliser avec STV_RECENTS pour identifier les requêtes qui s'exécutent actuellement ou qui ont abouti récemment.
- SYS_QUERY_HISTORY est utile pour la résolution des problèmes. Les requêtes DDL et DML y sont présentées avec des propriétés utiles telles que leur statut actuel, comme `running` ou `failed`, la durée d'exécution de chacune d'elles et si une requête s'est exécutée sur un cluster de mise à l'échelle de la simultanéité.
- STL_QUERYTEXT capture le texte de requête pour les commandes SQL. Par ailleurs, SVV_QUERY_INFLIGHT, qui joint STL_QUERYTEXT à STV_INFLIGHT, affiche des métadonnées de requête supplémentaires.
- Un conflit de verrous de transactions peut être à l'origine de problèmes de performances au niveau des requêtes. Pour en savoir plus sur les transactions qui maintiennent actuellement des verrous sur des tables, consultez SVV_TRANSACTIONS.
- La rubrique Identification des requêtes particulièrement indiquées pour un ajustement propose une requête de résolution des problèmes qui vous aide à identifier les requêtes exécutées récemment qui se sont montrées les plus chronophages. Elle vous permet ainsi de porter vos efforts sur les requêtes qui ont besoin d'être améliorées.
- Si vous souhaitez explorer de manière plus approfondie la gestion des requêtes et comprendre comment gérer les files d'attente de requêtes, la rubrique Gestion de la charge de travail vous

montre comment faire. La gestion de la charge de travail étant une fonctionnalité avancée, nous recommandons la gestion automatisée de la charge de travail dans la plupart des cas.

La charge échoue

Le chargement de vos données peut échouer pour les raisons suivantes. Nous vous suggérons les approches de résolution des problèmes suivantes.

La source de données se trouve dans une autre AWS région

Par défaut, le compartiment Amazon S3 ou la table Amazon DynamoDB spécifiés dans la commande COPY doivent se trouver dans la AWS même région que le cluster. Si vos données et votre cluster se trouvent dans des régions différentes, vous allez recevoir une erreur similaire à celle-ci :

```
The bucket you are attempting to access must be addressed using the specified endpoint.
```

Si possible, assurez-vous que votre cluster et votre source de données se trouvent dans la même région. Vous pouvez spécifier une autre région à l'aide de l'option REGION avec la commande COPY.

> **Note**
>
> Si votre cluster et votre source de données se trouvent dans des AWS régions différentes, des frais de transfert de données sont à votre charge. Vous avez également une latence plus élevée.

La commande COPY échoue

Interrogez STL_LOAD_ERRORS pour découvrir les erreurs qui se sont produites lors de charges spécifiques. Pour de plus amples informations, veuillez consulter STL_LOAD_ERRORS.

La charge est trop longue

Votre opération de chargement peut prendre trop de temps pour les raisons suivantes. Nous vous suggérons les approches de résolution des problèmes suivantes.

La commande COPY charge les données à partir d'un fichier unique

Fractionnez les données de chargement en plusieurs fichiers. Lorsque vous chargez toutes les données d'un seul grand fichier, Amazon Redshift est obligé d'effectuer un chargement sérialisé, ce qui est beaucoup plus lent. Le nombre de fichiers doit être un multiple du nombre de tranches de votre cluster, et les fichiers doivent être de taille égale, entre 1 Mo et 1 Go après compression. Pour de plus amples informations, veuillez consulter Bonnes pratiques Amazon Redshift pour la conception de requêtes.

L'opération de chargement utilise plusieurs commandes COPY

Si vous utilisez plusieurs commandes COPY simultanées pour charger une table à partir de plusieurs fichiers, Amazon Redshift est obligé d'effectuer un chargement sérialisé, ce qui est beaucoup plus lent. Dans ce cas, utilisez une seule commande COPY.

Le chargement des données est incorrect

Votre opération COPY peut charger des données incorrectes des manières suivantes. Nous vous suggérons les approches de résolution des problèmes suivantes.

Des fichiers incorrects sont chargés

L'utilisation d'un préfixe d'objet pour spécifier les fichiers de données peut entraîner la lecture de fichiers indésirables. Au lieu de cela, utilisez un fichier manifeste pour spécifier exactement les fichiers à charger. Pour plus d'informations, consultez l'option copy_from_s3_manifest_file pour la commande COPY et Example: COPY from Amazon S3 using a manifest dans les exemples COPY.

Définition du paramètre de taille d'extraction JDBC

Par défaut, le pilote JDBC collecte tous les résultats pour une requête à la fois. Par conséquent, lorsque vous tentez de récupérer un ensemble de résultats volumineux via une connexion JDBC, vous risquez de rencontrer une erreur côté client out-of-memory. Pour permettre à votre client de récupérer des ensembles de résultats par lots plutôt que par une seule all-or-nothing extraction, définissez le paramètre de taille de lecture JDBC dans votre application cliente.

> **Note**
>
> La taille de l'extraction n'est pas prise en charge pour ODBC.

Pour obtenir les meilleures performances, définissez la taille de l'extraction sur la valeur la plus élevée qui n'entraîne pas d'erreurs de mémoire. Une valeur d'extraction de taille inférieure entraîne

plusieurs opérations du serveur, ce qui prolonge les temps d'exécution. Le serveur réserve des ressources, y compris l'emplacement de requête WLM et la mémoire associée, jusqu'à ce que le client récupère le jeu de résultats complet ou que la requête soit annulée. Lorsque vous ajustez la taille d'extraction de façon appropriée, ces ressources sont libérées plus rapidement, ce qui les rend disponibles pour d'autres requêtes.

> **Note**
>
> Si vous devez extraire des ensembles de données volumineux, nous vous recommandons d'utiliser une instruction UNLOAD pour transférer les données vers Amazon S3. Lorsque vous utilisez UNLOAD, les nœuds de calcul fonctionnent en parallèle afin d'accélérer le transfert des données.

Pour plus d'informations sur la définition du paramètre de taille d'extraction JDBC, consultez Obtention de résultats basés sur un curseur dans la documentation de PostgreSQL.

Gestion de la charge de travail

Vous pouvez configurer Amazon Redshift WLM pour qu'il fonctionne en WLM automatique ou en WLM manuel.

Avec Amazon Redshift, vous pouvez gérer et hiérarchiser les requêtes simultanées et les charges de travail des utilisateurs afin d'optimiser les performances et l'utilisation des ressources. La gestion de la charge de travail (WLM) vous permet de définir des files d'attente, des groupes d'utilisateurs et d'autres structures pour contrôler les ressources allouées aux différents types de requêtes ou d'utilisateurs.

Les sections suivantes décrivent les fonctionnalités spécifiques de gestion de la charge de travail d'Amazon Redshift et vous guident dans leur configuration et leur surveillance.

Gestion automatique de la charge de travail

Pour maximiser le débit du système et utiliser les ressources efficacement, vous pouvez permettre à Amazon Redshift de gérer la façon dont les ressources sont divisées pour exécuter des requêtes simultanées avec WLM automatique. Le WLM automatique gère les ressources nécessaires à l'exécution des requêtes. Amazon Redshift détermine le nombre de requêtes exécutées simultanément et la quantité de mémoire allouée à chaque requête distribuée. Utilisez la gestion automatique de la charge de travail pour permettre à Amazon Redshift de gérer la façon dont les ressources sont divisées pour exécuter des requêtes simultanées. Pour de plus amples informations, veuillez consulter Implémentation de la gestion automatique de la charge de travail.

Avec la mise à l'échelle de la simultanéité et la gestion automatique de la charge de travail, vous pouvez prendre en charge un nombre pratiquement illimité d'utilisateurs simultanés et de requêtes simultanées, avec des performances de requêtes toujours rapides. Pour de plus amples informations, veuillez consulter Mise à l'échelle de la simultanéité.

> **Note**
>
> Dans la plupart des cas, nous vous recommandons d'utiliser le WLM automatique. Si vous utilisez le WLM manuel et que vous souhaitez passer du WLM automatique, consultez. Migration de la gestion manuelle de la charge de travail à la gestion automatique de la charge de travail

Avec Auto WLM, il est possible de définir les priorités des requêtes pour les charges de travail d'une file d'attente. Pour plus d'informations sur la priorité des requêtes, consultez Priorité de requête.

Gestion manuelle de la charge de travail

Vous pouvez avoir plusieurs sessions ou plusieurs utilisateurs peuvent exécuter des requêtes en même temps. Certaines requêtes peuvent consommer les ressources du cluster pendant de longues périodes et affecter les performances d'autres requêtes. Le WLM manuel peut aider à gérer cela pour des cas d'utilisation spécialisés. Utilisez la gestion manuelle de la charge de travail lorsque vous souhaitez plus de contrôle sur la simultanéité.

Vous pouvez gérer les performances du système en modifiant votre configuration WLM afin de créer des files d'attente distinctes pour les requêtes de longue durée et les requêtes de courte durée. Au moment de l'exécution, vous pouvez acheminer les requêtes vers ces files d'attente en fonction des groupes d'utilisateurs ou des groupes de requêtes.

Vous pouvez configurer des règles pour acheminer les requêtes vers des files d'attente spécifiques en fonction de l'utilisateur exécutant la requête ou d'étiquettes que vous spécifiez. Vous pouvez également configurer la quantité de mémoire allouée à chaque file d'attente, de telle sorte que les requêtes importantes s'exécutent dans des files d'attente disposant d'une mémoire supérieure aux autres files d'attente. Vous pouvez également configurer une règle de surveillance de requête (QRM) pour limiter les requêtes de longue durée. Pour de plus amples informations, veuillez consulter Implémentation de la gestion manuelle de la charge de travail.

> **Note**
>
> Nous vous recommandons de configurer vos files d'attente de requêtes WLM manuelles avec 15 emplacements de requête au maximum. Pour de plus amples informations, veuillez consulter Niveau de simultanéité.

Notez que dans le cas d'une configuration WLM manuelle, le nombre maximum d'emplacements que vous pouvez allouer à une file d'attente est de 50. Toutefois, cela ne signifie pas que dans une configuration WLM automatique, un cluster Amazon Redshift exécute toujours 50 requêtes simultanément. Cela peut changer en fonction des besoins en mémoire ou d'autres types d'allocation de ressources sur le cluster.

Rubriques

- Commutation du mode WLM

- Modifier la configuration WLM
- Implémentation de la gestion automatique de la charge de travail
- Implémentation de la gestion manuelle de la charge de travail
- Mise à l'échelle de la simultanéité
- Accélération des requêtes courtes
- Règles d'affectation de file d'attente de WLM
- Affectation des requêtes à des files d'attente
- Propriétés de configuration dynamiques et statiques WLM
- Règles de surveillance de requête WLM
- Tables et vues système WLM

Commutation du mode WLM

Vous pouvez activer le WLM automatique ou manuel à l'aide de la console Amazon Redshift :

1. Choisissez Switch WLM mode (Changer le mode de gestion de la charge de travail).
2. Pour le configurer sur WLM automatique, choisissez Auto WLM. En mode automatique, huit files d'attente maximum permettent de gérer les requêtes, et les champs Memory (Mémoire) et Concurrency on main (Simultanéité sur cluster principal) sont tous deux définis sur Auto. En outre, la priorité par défaut des requêtes est définie sur Normal.
3. Pour activer la configuration manuelle à l'aide de la console Amazon Redshift, passez au mode WLM manuel. En mode manuel, vous spécifiez les files d'attente utilisées pour gérer les requêtes, ainsi que les valeurs des champs Mémoire et Concurrency on main (Simultanéité sur cluster principal). Avec la configuration manuelle, vous pouvez configurer jusqu'à huit files d'attente de requêtes et définir le nombre de requêtes qui peuvent s'exécuter dans chacune de ces files d'attente simultanément.

Modifier la configuration WLM

La solution la plus simple pour modifier la configuration WLM consiste à utiliser la console Amazon Redshift. Vous pouvez également utiliser l'API AWS CLI ou l'API Amazon Redshift.

Lorsque vous basculez votre cluster de la gestion automatique de la charge de travail à la gestion manuelle de la charge de travail, ce dernier passe à l'état `pending reboot`. Cette modification prend seulement effet lors du prochain redémarrage d'un cluster.

Pour plus d'informations sur la modification des configurations WLM, consultez Configuration de la gestion de la charge de travail dans le Guide de la gestion du cluster Amazon Redshift.

Migration de la gestion manuelle de la charge de travail à la gestion automatique de la charge de travail

Pour optimiser le débit du système et utiliser efficacement les ressources, nous vous recommandons de définir la gestion automatique de la charge de travail pour vos files d'attente. Utilisez l'approche suivante pour configurer une transition en douceur entre la gestion manuelle de la charge de travail et la gestion automatique de la charge de travail.

Pour passer de la gestion manuelle de la charge de travail à la gestion automatique de la charge de travail, tout en utilisant les priorités de requête, nous vous recommandons de créer un nouveau groupe de paramètres, puis d'attacher ce groupe de paramètres à votre cluster. Pour plus d'informations, consultez Groupes de paramètres Amazon Redshift dans le Guide de gestion Amazon Redshift.

> **Important**
>
> La modification du groupe de paramètres ou le passage de la gestion manuelle de la charge de travail à la gestion automatique de la charge de travail exige un redémarrage du cluster. Pour de plus amples informations, veuillez consulter Propriétés de configuration dynamiques et statiques WLM.

Prenons un exemple avec trois files d'attente de gestion manuelle de la charge de travail. Une pour une charge de travail ETL, une pour une charge de travail d'analyse et une pour une charge de travail de science des données. La charge de travail ETL s'exécute toutes les 6 heures, la charge de travail d'analyse s'exécute tout au long de la journée et la charge de travail de science des données peut connaître des pics à tout moment. La gestion manuelle de la charge de travail vous permet de spécifier la mémoire et la simultanéité de chaque file d'attente de charge de travail, en fonction de votre compréhension de l'importance de chaque charge de travail pour l'entreprise. La spécification de la mémoire et de la simultanéinté est non seulement difficile à comprendre, mais elle se traduit

également par le partitionnement statique des ressources du cluster et par conséquent, par leur perte lorsque seul un sous-ensemble des charges de travail s'exécute.

Vous pouvez utiliser une gestion automatique de la charge de travail avec des priorités de requête pour indiquer les priorités relatives des charges de travail, tout en évitant les problèmes précédents. Pour cet exemple, procédez comme suit :

- Créez un nouveau groupe de paramètres et passez en mode Auto WLM (Gestion automatique de la charge de travail).
- Ajoutez des files d'attente à chacune des trois charges de travail : charge de travail ETL, charge de travail d'analyse et charge de travail de sciences de données. Utilisez les mêmes groupes d'utilisateurs pour chaque charge de travail que ceux utilisés en mode Gestion manuelle de la charge de travail.
- Définissez la priorité sur `High` pour la charge de travail ETL, sur `Normal` pour la charge de travail d'analyse, et sur `Low` pour la charge de travail de science des données. Ces priorités reflètent vos priorités métier pour les différents groupes d'utilisateurs ou charges de travail.
- Vous pouvez également activer la mise à l'échelle de la simultanéité pour la file d'attente d'analyse ou de science des données, afin que les performances des requêtes de ces files d'attente soient cohérentes même lorsque la charge de travail ETL est exécutée toutes les 6 heures.

Avec les priorités de requête, lorsque seule la charge de travail analytique s'exécute sur le cluster, elle bénéficie de l'intégralité des capacités du système. Cela permet d'obtenir un débit élevé et une meilleure utilisation du système. Toutefois, lorsque la charge de travail ETL démarre, elle est prioritaire en raison de sa priorité élevée. En plus de bénéficier d'une allocation préférentielle des ressources après avoir été admises, les requêtes s'exécutant dans le cadre de la charge de travail ETL sont prioritaires pendant l'admission. Ainsi, la charge de travail ETL s'exécute de manière prévisible quelles que soient les autres exécutions sur le système. Les performances prévisibles d'une charge de travail à priorité élevée s'effectuent au prix de charges de travail à priorité plus faible qui s'exécutent plus longtemps, car leurs requêtes attendent que des requêtes plus importantes se terminent. Ou, elles s'exécutent plus longtemps car elles récupèrent moins de ressources lorsqu'elles s'exécutent simultanément avec des requêtes à priorité plus élevée. Les algorithmes d'ordonnancement utilisés par Amazon Redshift facilitent le fait que les requêtes de moindre priorité ne souffrent pas de famine, mais continuent à progresser, bien qu'à un rythme plus lent.

Note

- Le champ Timeout (Expiration) n'est pas disponible en gestion automatique de la charge de travail. À la place, utilisez la règle QMR, `query_execution_time`. Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.
- L'action QMR, HOP, n'est pas applicable à la gestion automatique de la charge de travail. À la place, utilisez l'action `change priority`. Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.
- Les clusters utilisent différemment les files d'attente WLM automatiques et manuelles, ce qui peut prêter à confusion avec vos configurations. Par exemple, vous pouvez configurer la propriété de priorité dans les files d'attente WLM automatiques mais pas dans les files d'attente WLM manuelles. De ce fait, évitez de mélanger les files d'attente WLM automatiques et manuelles au sein d'un groupe de paramètres. À la place, créez un nouveau groupe de paramètres lors du passage à la gestion automatique de la charge de travail.

Implémentation de la gestion automatique de la charge de travail

En mode de gestion automatique de la charge de travail, Amazon Redshift gère la simultanéité des requêtes et l'allocation de mémoire. Vous pouvez créer jusqu'à huit files d'attente avec les identificateurs de classe de service 100–107. Chaque file d'attente a une priorité. Pour de plus amples informations, veuillez consulter Priorité de requête.

La gestion automatique de la charge de travail détermine la quantité de ressources nécessaires pour les requêtes et ajuste la simultanéité en fonction de la charge de travail. Lorsque des requêtes dans le système nécessitent une grande quantité de ressources (par exemple, des jointures de hachage entre des tables volumineuses), la simultanéité est plus faible. Lorsque des requêtes nécessitant moins de ressources (par exemple des insertions, des suppressions, des analyses ou des agrégations simples) sont soumises, la simultanéité est plus élevée.

La gestion automatique de la charge de travail est différente de l'accélération des requêtes courtes (SQA) et évalue différemment les requêtes. La gestion automatique de la charge de travail et l'accélération des requêtes courtes collaborent pour autoriser les requêtes légères à exécution rapide à être prises en charge même lorsque des requêtes lourdes nécessitant beaucoup de ressources sont en court d'exécution. Pour plus d'informations sur SQA, consultez Accélération des requêtes courtes.

Amazon Redshift active la gestion automatique de la charge de travail via des groupes de paramètres :

- Si vos clusters utilisent le groupe de paramètres par défaut, Amazon Redshift active la gestion automatique de la charge de travail pour eux.
- Si vos clusters utilisent des groupes de paramètres personnalisés, vous pouvez les configurer de manière à activer la gestion automatique de la charge de travail. Nous vous recommandons de créer un groupe de paramètres séparés pour votre configuration de gestion automatique de la charge de travail.

Pour configurer la gestion de la charge de travail, modifiez le paramètre (`wlm_json_configuration`) dans un groupe de paramètres qui peut être associé à un ou plusieurs clusters. Pour de plus amples informations, veuillez consulter Modifier la configuration WLM.

Vous définissez des files d'attente de requête au sein de la configuration WLM. Vous pouvez ajouter des files d'attente de requête supplémentaires à la configuration WLM par défaut (huit files d'attente de l'utilisateur au maximum). Vous pouvez configurer les éléments suivants pour chaque file d'attente de requête :

- Priorité
- Mode de mise à l'échelle de la simultanéité
- Groupes d'utilisateurs
- Groupes de requêtes
- Règles de surveillance de requête

Priorité

Vous pouvez définir l'importance relative des requêtes dans une charge de travail en définissant une valeur de priorité. La priorité est spécifiée pour une file d'attente et héritée par toutes les requêtes associées à la file d'attente. Pour de plus amples informations, veuillez consulter Priorité de requête.

Mode de mise à l'échelle de la simultanéité

Lorsque la mise à l'échelle de la simultanéité est activée, Amazon Redshift ajoute automatiquement de la capacité de cluster supplémentaire lorsque vous en avez besoin pour traiter une augmentation

des requêtes de lecture et d'écriture simultanées. Vos utilisateurs voient les données les plus récentes, que les requêtes soient exécutées sur le cluster principal ou sur un cluster de mise à l'échelle de la simultanéité.

Vous gérez quelles sont les requêtes envoyées au cluster de mise à l'échelle de la simultanéité en configurant les files d'attente de la gestion de la charge de travail. Lorsque vous activez la mise à l'échelle de la simultanéité pour une file d'attente, les requêtes éligibles sont envoyées au cluster de mise à l'échelle de la simultanéité au lieu d'être mises dans une file d'attente. Pour de plus amples informations, veuillez consulter Mise à l'échelle de la simultanéité.

Groupes d'utilisateurs

Vous pouvez affecter un ensemble de groupes d'utilisateurs à une file d'attente en spécifiant chaque nom de groupe d'utilisateurs ou en utilisant des caractères génériques. Lorsqu'un membre d'un groupe d'utilisateurs répertorié exécute une requête, celle-ci s'exécute dans la file d'attente correspondante. Il n'y a pas de limite au nombre de groupes d'utilisateurs qui peut être assigné à une file d'attente. Pour de plus amples informations, veuillez consulter Affectation des requêtes aux files d'attente en fonction des groupes d'utilisateurs.

Rôles utilisateurs

Vous pouvez attribuer un ensemble de rôles utilisateur à une file d'attente en spécifiant le nom de chaque rôle utilisateur ou en utilisant des caractères génériques. Lorsqu'un membre d'un rôle d'utilisateur répertorié exécute une requête, celle-ci s'exécute dans la file d'attente correspondante. Il n'existe aucune limite définie quant au nombre de rôles utilisateur pouvant être attribués à une file d'attente. Pour plus d'informations, consultez Affectation des requêtes aux files d'attente en fonction des rôles utilisateurs.

Groupes de requêtes

Vous pouvez affecter un ensemble de groupes de requêtes à une file d'attente en spécifiant chaque nom de groupe de requêtes ou en utilisant des caractères génériques. Un groupe de requêtes est simplement une étiquette. Au moment de l'exécution, vous pouvez affecter l'étiquette de groupe de requête à une série de requêtes. Toutes les requêtes qui sont affectées à un groupe de requêtes répertorié sont exécutées dans la file d'attente correspondante. Il n'y a pas de limite au nombre de groupes de requêtes qui peut être assigné à une file d'attente. Pour de plus amples informations, veuillez consulter Affectation d'une requête à un groupe de requêtes.

Caractères génériques

Si les caractères génériques sont activés dans la configuration de file d'attente WLM, vous pouvez affecter des groupes d'utilisateurs et des groupes de requêtes à une file d'attente, soit individuellement, soit en utilisant des caractères génériques de type shell Unix. Le modèle correspondant est sensible à la casse.

Par exemple, le caractère générique « * » correspond à n'importe quel nombre de caractères. Ainsi, si vous ajoutez `dba_*` à la liste de groupes d'utilisateurs pour une file d'attente, toute requête exécutée par un utilisateur et appartenant à un groupe dont le nom commence par `dba_` est affectée à cette file d'attente. Exemples : `dba_admin` ou `DBA_primary`. Le caractère générique « ? » correspond à n'importe quel caractère unique. Ainsi, si la file d'attente comprend le groupe utilisateur `dba?1`, alors les groupes utilisateur nommés `dba11` et `dba21` correspondent, mais le groupe `dba12` ne correspond pas.

Par défaut, les caractères génériques sont désactivés.

Règles de surveillance de requête

Les règles de surveillance de requête définissent les limites de performance reposant sur les métriques pour les files d'attente WLM et spécifient quelle action exécuter quand une requête dépasse ces limites. Par exemple, pour une file d'attente dédiée aux requêtes de courte durée, vous pouvez créer une règle qui annule les requêtes qui s'exécutent pendant plus de 60 secondes. Si vous souhaitez effectuer un suivi des requêtes mal conçues, vous pouvez configurer une autre règle qui consigne les requêtes contenant des boucles imbriquées. Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.

Vérification de la gestion automatique de la charge de travail

Pour vérifier si la gestion automatique de la charge de travail est activée, exécutez la requête suivante. Si la requête renvoie au moins une ligne, cela signifie que la gestion automatique de la charge de travail est activée.

```
select * from stv_wlm_service_class_config
where service_class >= 100;
```

La requête suivante affiche le nombre de requêtes ayant parcouru chaque file d'attente de requête (classe de service). Elle affiche aussi la durée d'exécution moyenne, le nombre de requêtes avec un

délai d'attente au 90e percentile et le délai d'attente moyen. Les requêtes WLM automatiques utilisent les classes de service 100 à 107.

```
select final_state, service_class, count(*), avg(total_exec_time),
percentile_cont(0.9) within group (order by total_queue_time), avg(total_queue_time)
from stl_wlm_query where userid >= 100 group by 1,2 order by 2,1;
```

Pour savoir quelles requêtes ont été exécutées par la gestion automatique de la charge de travail et exécutées avec succès, exécutez la requête suivante.

```
select a.queue_start_time, a.total_exec_time, label, trim(querytxt)
from stl_wlm_query a, stl_query b
where a.query = b.query and a.service_class >= 100 and a.final_state = 'Completed'
order by b.query desc limit 5;
```

Priorité de requête

Avec Amazon Redshift, vous pouvez gérer la priorisation des requêtes et l'allocation des ressources entre les requêtes et les charges de travail simultanées à l'aide de la gestion de la charge de travail (WM). Les sections suivantes expliquent comment configurer les files d'attente WM, définir les propriétés des files d'attente telles que l'allocation de mémoire et le dimensionnement de la simultanéité, et implémenter des règles de priorité adaptées aux exigences de votre charge de travail.

Toutes les requêtes n'ont pas la même importance, et souvent les performances d'une charge de travail ou d'un ensemble d'utilisateurs peuvent être plus importantes. Si vous avez activé la gestion automatique de la charge de travail, vous pouvez définir l'importance relative des requêtes dans une charge de travail en définissant une valeur de priorité. La priorité est spécifiée pour une file d'attente et héritée par toutes les requêtes associées à la file d'attente. Vous associez des requêtes à une file d'attente en mappant des groupes d'utilisateurs et des groupes de requêtes sur la file d'attente. Vous pouvez définir les priorités suivantes (de la plus élevée à la plus faible) :

1. `HIGHEST`
2. `HIGH`
3. `NORMAL`
4. `LOW`
5. `LOWEST`

Les administrateurs utilisent ces priorités pour montrer l'importance relative de leurs charges de travail lorsque des requêtes aux priorités différentes sont liées aux mêmes ressources. Amazon Redshift utilise la priorité lorsqu'il laisse des requêtes dans le système et pour déterminer la quantité de ressources allouées à une requête. Par défaut, les requêtes s'exécutent avec leur priorité définie sur `NORMAL`.

Une priorité supplémentaire, `CRITICAL`, qui est une priorité plus élevée que `HIGHEST`, est disponible pour les super-utilisateurs. Pour définir cette priorité, vous pouvez utiliser les fonctions CHANGE_QUERY_PRIORITY, CHANGE_SESSION_PRIORITY, et CHANGE_USER_PRIORITY. Pour autoriser un utilisateur de base de données à utiliser ces fonctions, vous pouvez créer une procédure stockée et accorder l'autorisation à un utilisateur. Pour obtenir un exemple, consultez CHANGE_SESSION_PRIORITY.

> **Note**
>
> Seule une requête `CRITICAL` peut être exécutée à la fois.
> Les annulations sont toujours exécutées en tant que priorité CRITIQUE.

Prenons un exemple dans lequel la priorité d'une charge de travail d'extraction, de transformation et de chargement (ETL) est supérieure à la priorité de la charge de travail d'analyse. La charge de travail ETL s'exécute toutes les 6 heures, et la charge de travail d'analyse s'exécute tout au long de la journée. Lorsque seule la charge de travail d'analyse s'exécute sur le cluster, les priorités de requête permettent au système de générer un haut débit avec une utilisation optimale du système. Toutefois, lorsque la charge de travail ETL démarre, elle est prioritaire en raison de sa priorité élevée. En plus de bénéficier d'une allocation préférentielle des ressources après avoir été admises, les requêtes s'exécutant dans le cadre de la charge de travail ETL sont prioritaires pendant l'admission. Ainsi, la charge de travail ETL s'exécute de manière prévisible quelles que soient les autres exécutions sur le système. Il fournit ainsi des performances prévisibles et permet aux administrateurs de fournir des accords de niveau de service (SLAs) à leurs utilisateurs professionnels.

Au sein d'un cluster donné, les performances prévisibles d'une charge de travail à priorité élevée s'effectuent au prix de charges de travail à priorité plus faible. Les charges de travail à priorité plus faible s'exécutent plus longtemps, car leurs requêtes attendent que des requêtes plus importantes se terminent. Ou, elles s'exécutent plus longtemps car elles récupèrent moins de ressources lorsqu'elles s'exécutent simultanément avec des requêtes à priorité plus élevée. Les requêtes à priorité plus faible ne connaissent pas de pénurie, mais poursuivent leur progression à un rythme inférieur.

Dans l'exempleprécédent, l'administrateur peut activer mise à l'échelle de la simultanéité pour la charge de travail d'analyse. Ainsi, cela permet à cette charge de travail de conserver sont débit, même si la charge de travail ETL s'exécute à une priorité élevée.

Configuration de la priorité de file d'attente

Si vous avez activé la gestion automatique de la charge de travail, chaque file d'attente a une valeur de priorité. Les requêtes sont acheminées vers les files d'attente en fonction des groupes d'utilisateurs et des groupes de requêtes. Commencez avec une priorité de file d'attente définie sur `NORMAL`. Définissez la priorité plus élevée ou plus faible en fonction de la charge de travail associée aux groupes d'utilisateurs et groupes de requêtes de la file d'attente.

Vous pouvez modifier la priorité d'une file d'attente sur la console Amazon Redshift. Sur la console Amazon Redshift, la page Gestion de la charge de travail affiche les files d'attente et permet de modifier les propriétés des files d'attente telles que la priorité. Pour définir la priorité à l'aide de la CLI ou les opérations d'API, utilisez le paramètre `wlm_json_configuration`. Pour plus d'informations, consultez Configuration de la gestion de la charge de travail dans le Guide de la gestion du cluster Amazon Redshift.

L'exemple suivant `wlm_json_configuration` définit trois groupes d'utilisateurs (`ingest`, `reporting` et `analytics`). Les requêtes envoyées par les utilisateurs de l'un de ces groupes s'exécutent avec respectivement les priorités `highest`, `normal` et `low`.

```
[
    {
        "user_group": [
            "ingest"
        ],
        "priority": "highest",
        "queue_type": "auto"
    },
    {
        "user_group": [
            "reporting"
        ],
        "priority": "normal",
        "queue_type": "auto"
    },
    {
        "user_group": [
            "analytics"
```

```
        ],
        "priority": "low",
        "queue_type": "auto",
        "auto_wlm": true
    }
]
```

Modification de la priorité de requête avec les règles de surveillance de requête

Les règles de surveillance de requête (QMR) vous permettent de modifier la priorité d'une requête en fonction de son comportement pendant qu'elle s'exécute. Pour ce faire, spécifiez l'attribut de priorité dans une action et un prédicat QMR. Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.

Par exemple, vous pouvez définir une règle pour annuler n'importe quelle requête classée en tant que priorité `high` qui s'exécute pendant plus de 10 minutes.

```
"rules" :[
  {
    "rule_name":"rule_abort",
    "predicate":[
      {
        "metric_name":"query_cpu_time",
        "operator":">",
        "value":600
      },
      {
        "metric_name":"query_priority",
        "operator":"=",
        "value":"high"
      }
    ],
    "action":"abort"
  }
]
```

Un autre exemple consiste à spécifier une règle pour définir la priorité de requête sur `lowest` pour n'importe quelle priorité `normal` actuelle qui déverse plus de 1 To sur un disque.

```
"rules":[
  {
    "rule_name":"rule_change_priority",
```

```
    "predicate":[
      {
        "metric_name":"query_temp_blocks_to_disk",
        "operator":">",
        "value":1000000
      },
      {
        "metric_name":"query_priority",
        "operator":"=",
        "value":"normal"
      }
    ],
    "action":"change_query_priority",
    "value":"lowest"
  }
]
```

Surveillance de la priorité de requête

Pour afficher la priorité pour les requêtes en attente et en cours d'exécution, consultez la colonne `query_priority` dans la table du système stv_wlm_query_state.

```
query    | service_cl | wlm_start_time             | state            | queue_time |
 query_priority
---------+------------+----------------------------+------------------+------------
+----------------
2673299  | 102        | 2019-06-24 17:35:38.866356 | QueuedWaiting    | 265116     |
 Highest
2673236  | 101        | 2019-06-24 17:35:33.313854 | Running          | 0          |
 Highest
2673265  | 102        | 2019-06-24 17:35:33.523332 | Running          | 0          |
 High
2673284  | 102        | 2019-06-24 17:35:38.477366 | Running          | 0          |
 Highest
2673288  | 102        | 2019-06-24 17:35:38.621819 | Running          | 0          |
 Highest
2673310  | 103        | 2019-06-24 17:35:39.068513 | QueuedWaiting    | 62970      |
 High
2673303  | 102        | 2019-06-24 17:35:38.968921 | QueuedWaiting    | 162560     |
 Normal
2673306  | 104        | 2019-06-24 17:35:39.002733 | QueuedWaiting    | 128691     |
 Lowest
```

Pour répertorier les priorités de requête pour les requêtes terminées, consultez la colonne `query_priority` dans la table de système stl_wlm_query.

```
select query, service_class as svclass, service_class_start_time as starttime,
 query_priority
from stl_wlm_query order by 3 desc limit 10;
```

```
  query   | svclass |         starttime          |    query_priority
---------+---------+----------------------------+----------------------
 2723254 |     100 | 2019-06-24 18:14:50.780094 | Normal
 2723251 |     102 | 2019-06-24 18:14:50.749961 | Highest
 2723246 |     102 | 2019-06-24 18:14:50.725275 | Highest
 2723244 |     103 | 2019-06-24 18:14:50.719241 | High
 2723243 |     101 | 2019-06-24 18:14:50.699325 | Low
 2723242 |     102 | 2019-06-24 18:14:50.692573 | Highest
 2723239 |     101 | 2019-06-24 18:14:50.668535 | Low
 2723237 |     102 | 2019-06-24 18:14:50.661918 | Highest
 2723236 |     102 | 2019-06-24 18:14:50.643636 | Highest
```

Pour optimiser le débit de votre charge de travail, Amazon Redshift peut modifier la priorité des requêtes soumises par l'utilisateur. Amazon Redshift utilise des algorithmes de machine learning avancés pour déterminer quand cette optimisation profite à votre charge de travail et l'applique automatiquement lorsque toutes les conditions suivantes sont remplies.

- La gestion automatique de la charge de travail (WLM) est activée.
- Une seule file d'attente WLM est définie.
- Vous n'avez pas défini de règles de surveillance des requêtes (QMRs) qui définissent la priorité des requêtes. Ces règles comprennent la métrique QMR `query_priority` ou l'action QMR `change_query_priority`. Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.

Implémentation de la gestion manuelle de la charge de travail

Grâce à la gestion manuelle de la charge de travail, vous pouvez gérer les performances du système et l'expérience de vos utilisateurs en modifiant la configuration de gestion de la charge de travail afin de créer des files d'attente distinctes pour les requêtes de longue durée et de courte durée.

Lorsque les utilisateurs exécutent des requêtes dans Amazon Redshift, celles-ci sont acheminées vers des files d'attente de requêtes. Chaque file d'attente de requête contient un certain nombre d'emplacements de requête. Chaque file d'attente se voit affecter une partie de la mémoire disponible du cluster. La mémoire d'une file d'attente est répartie entre les emplacements de requête de la file d'attente. Vous pouvez permettre à Amazon Redshift de gérer la concurrence des requêtes en WLM automatique. Pour de plus amples informations, veuillez consulter Implémentation de la gestion automatique de la charge de travail.

Ou vous pouvez configurer des propriétés de gestion de la charge de travail pour chaque file d'attente de requête. Ainsi, vous spécifiez la façon dont la mémoire est allouée entre les emplacements et comment les requêtes peuvent être acheminées vers des files d'attente spécifiques lors de l'exécution. Vous pouvez également configurer les propriétés de gestion de la charge de travail afin d'annuler les requêtes de longue durée.

Par défaut, Amazon Redshift configure les files d'attente de requêtes suivantes :

- Une file d'attente de super-utilisateur

  La file d'attente de super-utilisateur est réservée aux super-utilisateurs uniquement et ne peut pas être configurée. Utilisez cette file d'attente uniquement lorsque vous avez besoin d'exécuter des requêtes qui affectent le système ou à des fins de dépannage. Par exemple, utilisez cette file d'attente lorsque vous avez besoin d'annuler une requête de longue durée d'un utilisateur ou d'ajouter des utilisateurs à la base de données. Ne l'utilisez pas pour exécuter des requêtes courantes. La file d'attente ne s'affiche pas dans la console, mais elle s'affiche dans les tables système de la base de données comme la cinquième file d'attente. Pour exécuter une requête dans la file d'attente du super-utilisateur, un utilisateur doit être connecté en tant que super-utilisateur et exécuter la requête à l'aide du groupe de requêtes `superuser` prédéfini.
- Une file d'attente d'utilisateur par défaut

  La file d'attente par défaut est configurée initialement pour exécuter cinq requêtes simultanément. Lorsque vous utilisez le WLM manuel, vous pouvez modifier les propriétés de simultanéité, d'expiration et d'allocation de mémoire de la file d'attente par défaut, mais vous ne pouvez pas spécifier de groupes d'utilisateurs ou de groupes de requêtes. La file d'attente par défaut doit être la dernière file d'attente de la configuration de la charge de travail. Toutes les requêtes qui ne sont pas acheminées vers d'autres files d'attente s'exécutent dans la file d'attente par défaut.

Les files d'attente de requêtes sont définies dans la configuration WLM. La configuration WLM est un paramètre modifiable (`wlm_json_configuration`) dans un groupe de paramètres, qui peut être

associé à un ou plusieurs clusters. Pour plus d'informations, consultez Configuration de la gestion de la charge de travail dans le Guide de la gestion du cluster Amazon Redshift.

Vous pouvez ajouter des files d'attente de requête supplémentaires à la configuration WLM par défaut (huit files d'attente de l'utilisateur au maximum). Vous pouvez configurer les éléments suivants pour chaque file d'attente de requête :

- Mode de mise à l'échelle de la simultanéité
- Niveau de simultanéité
- Groupes d'utilisateurs
- Groupes de requêtes
- Pourcentage de mémoire WLM à utiliser
- Délai WLM
- Saut de file d'attente des requêtes WLM
- Règles de surveillance de requête

Mode de mise à l'échelle de la simultanéité

Lorsque la mise à l'échelle de la simultanéité est activée, Amazon Redshift ajoute automatiquement de la capacité de cluster supplémentaire lorsque vous en avez besoin pour traiter une augmentation des requêtes de lecture et d'écriture simultanées. Les utilisateurs voient les données les plus récentes, que les requêtes s'exécutent sur le cluster principal ou sur un cluster de mise à l'échelle de la simultanéité.

Vous gérez quelles sont les requêtes envoyées au cluster de mise à l'échelle de la simultanéité en configurant les files d'attente de la gestion de la charge de travail. Lorsque vous activez la mise à l'échelle de la simultanéité pour une file d'attente, les requêtes éligibles sont envoyées au cluster de mise à l'échelle de la simultanéité au lieu d'être mises dans une file d'attente. Pour de plus amples informations, veuillez consulter Mise à l'échelle de la simultanéité.

Niveau de simultanéité

Les requêtes placées dans une file d'attente s'exécutent simultanément jusqu'à ce qu'elles atteignent le nombre d'emplacements de requêtes WLM ou le niveau de simultanéité défini pour cette file d'attente. Les requêtes suivantes attendent ensuite dans la file d'attente.

> **Note**
>
> Le niveau de simultanéité est différent du nombre de connexions utilisateur simultanées qu'un cluster peut accepter. Pour plus d'informations, consultez Connexion à un cluster dans le Guide de la gestion du cluster Amazon Redshift.

Dans une configuration WLM automatique, ce qui est recommandé, le niveau de simultanéité est réglé sur Auto. Amazon Redshift alloue dynamiquement de la mémoire aux requêtes, ce qui détermine par la suite le nombre de requêtes à exécuter simultanément. Il se base sur les ressources requises pour les requêtes en cours et les requêtes en file d'attente. La fonction Auto WLM n'est pas configurable. Pour de plus amples informations, veuillez consulter Implémentation de la gestion automatique de la charge de travail.

Dans une configuration WLM manuelle, Amazon Redshift alloue statiquement une quantité fixe de mémoire à chaque file d'attente. La mémoire de la file d'attente est répartie de manière égale entre les emplacements de requête. Par exemple, si une file d'attente se voit attribuer 20 % de la mémoire d'un cluster et dispose de 10 emplacements, chaque requête se voit attribuer 2 % de la mémoire du cluster. L'allocation de mémoire reste fixe quel que soit le nombre de requêtes exécutées simultanément. En raison de cette allocation de mémoire fixe, les requêtes qui s'exécutent entièrement en mémoire lorsque le nombre d'emplacements est de 5 peuvent écrire les résultats intermédiaires sur le disque si le nombre d'emplacements est porté à 20. Dans ce cas, la part de chaque requête dans la mémoire de la file d'attente passe de 1/5e à 1/20e. Le disque supplémentaire est I/O susceptible de dégrader les performances.

Le nombre maximum d'emplacements pour toutes les files d'attente définies par l'utilisateur est de 50. Cela limite le nombre total d'emplacements pour toutes les files d'attente, y compris la file d'attente par défaut. La seule file d'attente qui n'est pas soumise à cette limite est la file d'attente réservée au super-utilisateur.

Par défaut, les files d'attente WLM manuelles ont un niveau de simultanéité de 5. Votre charge de travail peut tirer profit d'un niveau de simultanéité plus élevé dans certains cas, notamment :

- Si plusieurs petites requêtes sont obligées d'attendre les requêtes de longue durée, créez une file d'attente distincte avec un nombre d'emplacements supérieur et affectez les requêtes plus petites à cette file d'attente. Une file d'attente avec un niveau de simultanéité supérieur a moins de mémoire allouée à chaque emplacement de requête, mais les requêtes plus petites nécessitent moins de mémoire.

> **Note**
>
> Si vous activez l'accélération des requêtes courtes (SQA), WLM établit automatiquement la priorité des requêtes de courte durée sur les requêtes de longue durée, afin que vous n'ayez pas besoin d'une file d'attente distincte pour les requêtes courtes dans la plupart des flux de travail. Pour de plus amples informations, veuillez consulter Accélération des requêtes courtes.

- Si plusieurs de vos requêtes accèdent chacune aux données sur une seule tranche, configurez une file d'attente WLM distincte pour exécuter ces requêtes simultanément. Amazon Redshift affecte des requêtes simultanées à des tranches distinctes, ce qui permet à plusieurs requêtes d'être exécutées en parallèle sur plusieurs tranches. Par exemple, si une requête est un agrégat simple avec un prédicat sur la clé de distribution, les données de la requête sont situées sur une seule tranche.

Exemple de WLM manuel

Cet exemple est un scénario WLM manuel simple qui montre comment les emplacements et la mémoire peuvent être alloués. Vous mettez en œuvre un WLM manuel avec trois files d'attente, qui sont les suivantes :

- File d'attente d'ingestion de données : cette file d'attente est configurée pour l'ingestion de données. Elle se voit allouer 20 % de la mémoire du cluster et dispose de 5 emplacements. Par la suite, 5 requêtes peuvent être exécutées simultanément dans la file d'attente et chacune se voit attribuer 4 % de la mémoire.
- File d'attente spécialiste des données : cette file d'attente est conçue pour les requêtes nécessitant beaucoup de mémoire. Elle se voit allouer 40 % de la mémoire du cluster et dispose de 5 emplacements. Par la suite, 5 requêtes peuvent être exécutées simultanément et chacune se voit attribuer 8 % de la mémoire.
- File d'attente par défaut : cette file d'attente est destinée à la majorité des utilisateurs de l'organisation. Il s'agit notamment des groupes de vente et de comptabilité qui formulent généralement des requêtes courtes ou moyennes peu complexes. Elle se voit attribuer 40 % de la mémoire du cluster et dispose de 40 emplacements. 40 requêtes peuvent être exécutées simultanément dans cette file d'attente, chaque requête se voyant allouer 1 % de la mémoire. Il s'agit du nombre maximum d'emplacements qui peuvent être alloués à cette file d'attente, car la limite est de 50 pour toutes les files d'attente.

Si vous utilisez le WLM automatique et que votre charge de travail nécessite l'exécution de plus de 15 requêtes en parallèle, nous vous recommandons d'activer la mise à l'échelle de la simultanéité. Cela est dû au fait que l'augmentation du nombre d'emplacements de requête au-dessus de 15 peut créer une contention pour les ressources système et limiter le débit global d'un seul cluster. Avec la mise à l'échelle de la simultanéité, vous pouvez exécuter des centaines de requêtes en parallèle jusqu'à un nombre configuré de clusters de mise à l'échelle de la simultanéité. Le nombre de clusters de mise à l'échelle de la simultanéité est contrôlé par max_concurrency_scaling_clusters. Pour plus d'informations sur la mise à l'échelle de la concurrence, consultez Mise à l'échelle de la simultanéité.

Pour de plus amples informations, veuillez consulter Amélioration des performances des requêtes.

Groupes d'utilisateurs

Vous pouvez affecter un ensemble de groupes d'utilisateurs à une file d'attente en spécifiant chaque nom de groupe d'utilisateurs ou en utilisant des caractères génériques. Lorsqu'un membre d'un groupe d'utilisateurs répertorié exécute une requête, celle-ci s'exécute dans la file d'attente correspondante. Il n'y a pas de limite au nombre de groupes d'utilisateurs qui peut être assigné à une file d'attente. Pour de plus amples informations, veuillez consulter Affectation des requêtes aux files d'attente en fonction des groupes d'utilisateurs.

Rôles utilisateurs

Vous pouvez attribuer un ensemble de rôles utilisateur à une file d'attente en spécifiant le nom de chaque rôle utilisateur ou en utilisant des caractères génériques. Lorsqu'un membre d'un rôle d'utilisateur répertorié exécute une requête, celle-ci s'exécute dans la file d'attente correspondante. Il n'existe aucune limite définie quant au nombre de rôles utilisateur pouvant être attribués à une file d'attente. Pour de plus amples informations, veuillez consulter Affectation des requêtes aux files d'attente en fonction des rôles utilisateurs.

Groupes de requêtes

Vous pouvez affecter un ensemble de groupes de requêtes à une file d'attente en spécifiant chaque nom de groupe de requêtes ou en utilisant des caractères génériques. Un groupe de requêtes est simplement une étiquette. Au moment de l'exécution, vous pouvez affecter l'étiquette de groupe de requête à une série de requêtes. Toutes les requêtes qui sont affectées à un groupe de requêtes répertorié sont exécutées dans la file d'attente correspondante. Il n'y a pas de limite au nombre de groupes de requêtes qui peut être assigné à une file d'attente. Pour de plus amples informations, veuillez consulter Affectation d'une requête à un groupe de requêtes.

Caractères génériques

Si les caractères génériques sont activés dans la configuration de file d'attente WLM, vous pouvez affecter des groupes d'utilisateurs et des groupes de requêtes à une file d'attente individuellement ou en utilisant des caractères génériques de style shell Unix. Le modèle correspondant est sensible à la casse.

Par exemple, le caractère générique « * » correspond à n'importe quel nombre de caractères. Ainsi, si vous ajoutez `dba_*` à la liste de groupes d'utilisateurs pour une file d'attente, toute requête exécutée par un utilisateur et appartenant à un groupe dont le nom commence par `dba_` est affectée à cette file d'attente. `dba_admin` ou `DBA_primary` sont des exemples. Le caractère générique « ? » correspond à n'importe quel caractère unique. Ainsi, si la file d'attente comprend le groupe utilisateur `dba?1`, alors les groupes utilisateur nommés `dba11` et `dba21` correspondent, mais le groupe `dba12` ne correspond pas.

Les caractères génériques sont désactivés par défaut.

Pourcentage de mémoire WLM à utiliser

Dans une configuration de gestion automatique de la charge de travail, le pourcentage de mémoire est défini sur **`auto`**. Pour de plus amples informations, veuillez consulter Implémentation de la gestion automatique de la charge de travail.

Dans une configuration de gestion manuelle de la charge de travail, vous pouvez définir le paramètre `WLM Memory Percent to Use` pour spécifier la quantité de mémoire disponible qui est allouée à une requête. Par défaut, chaque file d'attente définie par l'utilisateur se voit allouer une part égale de la mémoire disponible pour les requêtes définies par l'utilisateur. Par exemple, si vous avez quatre files d'attente définies par l'utilisateur, chaque file d'attente reçoit 25 % de la mémoire disponible. La file d'attente du super-utilisateur dispose de sa propre mémoire allouée et ne peut pas être modifiée. Pour modifier l'allocation, vous affectez un pourcentage entier de la mémoire à chaque file d'attente, à concurrence d'un total de 100 %. Toute mémoire non allouée est gérée par Amazon Redshift et peut être donnée temporairement à une file d'attente si celle-ci demande de la mémoire supplémentaire pour le traitement.

Par exemple, si vous configurez quatre files d'attente, vous pouvez allouer de la mémoire comme suit : 20 %, 30 %, 15 %, 15 %. Les 20 % restants ne sont pas alloués et sont gérés par le service.

Délai WLM

Le délai WLM (`max_execution_time`) est obsolète. Au lieu de cela, créez une règle de surveillance de requête (QRM) `query_execution_time` pour limiter le temps d'exécution écoulé d'une requête. Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.

Pour limiter le délai imparti aux requêtes dans une file d'attente WLM donnée, vous pouvez définir la valeur du délai WLM pour chaque file d'attente. Le paramètre timeout spécifie la durée, en millisecondes, pendant laquelle Amazon Redshift attend l'exécution d'une requête avant de l'annuler ou de la faire sauter. Le délai s'appuie sur la durée de l'exécution de la requête et n'inclut pas le temps d'attente dans une file d'attente.

WLM tente de replacer les instructions CREATE TABLE AS (CTAS) et les requêtes en lecture seule, telles que les instructions SELECT. Les requêtes ne pouvant être replacées sont annulées. Pour de plus amples informations, veuillez consulter Saut de file d'attente des requêtes WLM.

Le délai WLM ne s'applique pas à une requête qui a atteint l'état returning. Pour afficher l'état d'une requête, consultez la table système STV_WLM_QUERY_STATE. Les instructions COPY et les opérations de maintenance, telles que ALTER, ANALYZE et VACUUM, ne sont pas soumises au délai d'expiration du WLM.

La fonction de délai WLM est similaire au paramètre de configuration statement_timeout. La seule différence est que le paramètre de configuration `statement_timeout` s'applique à l'ensemble du cluster, alors que le délai WLM est spécifique à une seule file d'attente de la configuration WLM.

Si statement_timeout est également spécifié, la valeur la plus basse de statement_timeout et de délai WLM (max_execution_time) est utilisée.

Règles de surveillance de requête

Les règles de surveillance de requête définissent les limites de performance reposant sur les métriques pour les files d'attente WLM et spécifient quelle action exécuter quand une requête dépasse ces limites. Par exemple, pour une file d'attente dédiée aux requêtes de courte durée, vous pouvez créer une règle qui annule les requêtes qui s'exécutent pendant plus de 60 secondes. Si vous souhaitez effectuer un suivi des requêtes mal conçues, vous pouvez configurer une autre règle qui consigne les requêtes contenant des boucles imbriquées. Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.

Saut de file d’attente des requêtes WLM

Avec Amazon Redshift, vous pouvez gérer la simultanéité de la charge de travail et l'allocation des ressources en activant le saut dans les files d'attente de requêtes WLM (Workload Management). Cette fonctionnalité permet aux requêtes de « passer » temporairement d'une file d'attente assignée à une file d'attente plus prioritaire lorsque des ressources sont disponibles, améliorant ainsi les performances globales des requêtes et l'utilisation du système. Les sections suivantes fournissent des instructions détaillées sur la configuration et l'utilisation du saut de file d'attente de requêtes WLM dans Amazon Redshift.

Une requête peut être replacée à cause d’un délai WLM ou d’une action de saut de règle de surveillance de requête (QMR). Vous pouvez uniquement sauter des requêtes dans une configuration WLM manuelle.

Lorsqu’une requête est replacée, la gestion de la charge de travail tente d’acheminer la requête vers la file d’attente correspondante suivante en fonction des règles d’affectation de file d’attente de WLM. Si la requête ne correspond à aucune autre définition de file d’attente, elle est annulée. Elle n’est pas affectée à la file d’attente par défaut.

Actions de délai WLM

Le tableau suivant résume le comportement des différents types de requêtes soumis à un délai WLM.

Type de requête	Action
INSERT, UPDATE et DELETE	Annuler
Fonctions définies par l'utilisateur () UDFs	Annuler
UNLOAD	Annuler
COPY	Continuer l’exécution
Opérations de maintenance	Continuer l’exécution
Requêtes en lecture seule dans un état de `returning`	Continuer l’exécution
Requêtes en lecture seule dans un état de `running`	Réaffecter ou redémarrer

Type de requête	Action
CREATE TABLE AS (CTAS), SELECT INTO	Réaffecter ou redémarrer

Saut de file d'attente en raison d'un délai WLM

WLM replace les types de requêtes suivants quand ceux-ci expirent :

- Les requêtes en lecture seule, telles que les instructions SELECT qui sont dans un état WLM de `running`. Pour rechercher l'état WLM d'une requête, affichez la colonne STATE sur la table système STV_WLM_QUERY_STATE.
- Les instructions CREATE TABLE AS (CTAS). Le saut de file d'attente WLM prend en charge à la fois les instructions CTAS définies par l'utilisateur et celles générées par le système.
- Les instructions SELECT INTO.

Les requêtes qui ne sont pas soumises au délai WLM continuent de s'exécuter dans la file d'attente d'origine jusqu'à ce qu'elles soient terminées. Les types de requêtes suivants ne sont pas soumis au délai WLM :

- Les instructions COPY
- Opérations de maintenance, telles que ALTER, ANALYZE et VACUUM
- Les requêtes en lecture seule, telles que les instructions SELECT qui ont atteint un état WLM de `returning`. Pour rechercher l'état WLM d'une requête, affichez la colonne STATE sur la table système STV_WLM_QUERY_STATE.

Les requêtes qui ne sont pas admissibles au saut en fonction d'un délai WLM sont annulées quand elles expirent. Les types de requêtes suivants ne sont pas admissibles au saut en fonction d'un délai WLM :

- Les instructions SELECT, UPDATE et DELETE
- Les instructions UNLOAD
- Fonctions définies par l'utilisateur () UDFs

Délai WLM des requêtes réaffectées et redémarrées

Lorsqu'une requête est replacée et qu'aucune file d'attente correspondante n'est trouvée, celle-ci est annulée.

Lorsqu'une requête est replacée et qu'une file d'attente correspondante est trouvée, WLM tente de réaffecter celle-ci à la nouvelle file d'attente. Si une requête ne peut être réaffectée, elle est redémarrée dans la nouvelle file d'attente, comme décrit ci-après.

Une requête est affectée uniquement si toutes les conditions suivantes sont vraies :

- Une file d'attente correspondante a été trouvée.
- La nouvelle file d'attente dispose de suffisamment d'emplacements libres pour exécuter la requête. Une requête peut nécessiter plusieurs emplacements si le paramètre wlm_query_slot_count a été défini sur une valeur supérieure à 1.
- La nouvelle file d'attente dispose d'au moins autant de mémoire disponible que la requête en utilise actuellement.

Si la requête est réaffectée, celle-ci continue à s'exécuter dans la nouvelle file d'attente. Les résultats intermédiaires sont préservés, de sorte que l'impact sur le temps d'exécution total est minime.

Si la requête ne peut être réaffectée, elle est annulée et redémarrée dans la nouvelle file d'attente. Les résultats intermédiaires sont supprimés. La requête attend dans la file d'attente, puis commence à s'exécuter lorsque suffisamment d'emplacements sont disponibles.

Action de saut QMR

Le tableau suivant résume le comportement des différents types de requêtes soumis à une action de saut QMR.

Type de requête	Action
COPY	Continuer l'exécution
Opérations de maintenance	Continuer l'exécution
Fonctions définies par l'utilisateur () UDFs	Continuer l'exécution
UNLOAD	Réaffecter ou continuer l'exécution

Type de requête	Action
INSERT, UPDATE et DELETE	Réaffecter ou continuer l'exécution
Requêtes en lecture seule dans un état de `returning`	Réaffecter ou continuer l'exécution
Requêtes en lecture seule dans un état de `running`	Réaffecter ou redémarrer
CREATE TABLE AS (CTAS), SELECT INTO	Réaffecter ou redémarrer

Pour savoir si une requête qui a été replacée par QMR a été réaffectée, redémarrée ou annulée, interrogez la table de journal système STL_WLM_RULE_ACTION.

Action de saut QMR des requêtes réaffectées et redémarrées

Lorsqu'une requête est replacée et qu'aucune file d'attente correspondante n'est trouvée, celle-ci est annulée.

Lorsqu'une requête est replacée et qu'une file d'attente correspondante est trouvée, WLM tente de réaffecter celle-ci à la nouvelle file d'attente. Si une requête ne peut être réaffectée, elle est redémarrée dans la nouvelle file d'attente ou continue de s'exécuter dans la file d'attente d'origine, comme décrit ci-après.

Une requête est affectée uniquement si toutes les conditions suivantes sont vraies :

- Une file d'attente correspondante a été trouvée.
- La nouvelle file d'attente dispose de suffisamment d'emplacements libres pour exécuter la requête. Une requête peut nécessiter plusieurs emplacements si le paramètre wlm_query_slot_count a été défini sur une valeur supérieure à 1.
- La nouvelle file d'attente dispose d'au moins autant de mémoire disponible que la requête en utilise actuellement.

Si la requête est réaffectée, celle-ci continue à s'exécuter dans la nouvelle file d'attente. Les résultats intermédiaires sont préservés, de sorte que l'impact sur le temps d'exécution total est minime.

Si une requête ne peut être réaffectée, elle est redémarrée ou continue de s'exécuter dans la file d'attente d'origine. Si la requête est redémarrée, elle est annulée et redémarrée dans la nouvelle file d'attente. Les résultats intermédiaires sont supprimés. La requête attend dans la file d'attente, puis commence à s'exécuter lorsque suffisamment d'emplacements sont disponibles.

Didacticiel : Configuration des files d'attente de gestion manuelle de la charge de travail

Avec Amazon Redshift, vous pouvez configurer des files d'attente de gestion manuelle de la charge de travail (WLM) afin de hiérarchiser et d'allouer des ressources pour différents types de requêtes et d'utilisateurs. Les files d'attente WLM manuelles vous permettent de contrôler les paramètres de mémoire et de simultanéité pour des files d'attente spécifiques, afin de garantir que les charges de travail critiques reçoivent les ressources nécessaires tout en empêchant les requêtes de faible priorité de monopoliser le système. Les sections suivantes vous guident tout au long du processus de création et de configuration de files d'attente WLM manuelles dans Amazon Redshift afin de répondre à vos exigences en matière de gestion de charge de travail.

Présentation

Nous recommandons de configurer la gestion automatique des charges de travail (WLM) dans Amazon Redshift. Pour de plus amples informations sur la gestion automatique de la charge de travail, veuillez consulter Gestion de la charge de travail. Toutefois, si vous avez besoin de plusieurs files d'attente WLM, ce tutoriel vous guide à travers le processus de configuration de la gestion manuelle des charges de travail (WLM) dans Amazon Redshift. Cette configuration vous permet d'améliorer les performances des requêtes et l'allocation des ressources dans votre cluster.

Amazon Redshift achemine les requêtes des utilisateurs vers les files d'attente en vue de leur traitement. WLM définit la façon dont ces requêtes sont acheminées vers les files d'attente. Par défaut, Amazon Redshift possède deux files d'attente disponibles pour les requêtes : une pour les superutilisateurs et une pour les utilisateurs. La file d'attente des super-utilisateurs ne peut pas être configurée et ne peut traiter qu'une requête à la fois. Vous devez réserver cette file d'attente à des fins de dépannage uniquement. La file d'attente des utilisateurs peut traiter jusqu'à cinq requêtes à la fois, mais vous pouvez configurer ce nombre en modifiant le niveau de concurrence de la file d'attente si nécessaire.

Lorsque vous avez plusieurs utilisateurs exécutant des requêtes sur la base de données, vous pouvez considérer une autre configuration comme étant plus efficace. Par exemple, si certains utilisateurs exécutent des opérations gourmandes en ressources, telles que VACUUM, celles-ci

peuvent avoir un impact négatif sur les requêtes moins gourmandes, telles que les rapports. Vous pouvez envisager d'ajouter des files d'attente supplémentaires et de les configurer pour différentes charges de travail.

Durée estimée : 75 minutes

Coût estimé : 50 cents

Prérequis

Vous avez besoin d'un cluster Amazon Redshift, de l'exemple de base de données TICKIT et de l'outil client Amazon Redshift RSQL. Si vous ne les avez pas encore installés, consultez Guide de démarrage d'Amazon Redshift et Amazon Redshift RSQL.

Sections

- Section 1 : Comprendre le comportement de traitement de file d'attente par défaut
- Section 2 : Modification de la configuration de la file d'attente des requêtes WLM
- Section 3 : Acheminement des requêtes vers les files d'attente en fonction des groupes d'utilisateurs et des groupes de requêtes
- Section 4 : Utilisation de wlm_query_slot_count pour remplacer temporairement le niveau de concurrence d'une file d'attente
- Section 5 : Nettoyage de vos ressources

Section 1 : Comprendre le comportement de traitement de file d'attente par défaut

Avant de commencer à configurer le WLM manuel, il est utile de comprendre le comportement par défaut du traitement des files d'attente dans Amazon Redshift. Dans cette section, vous créez deux vues de base de données qui retournent des informations à partir de plusieurs tables système. Ensuite, vous exécutez quelques requêtes de test pour voir comment elles sont routées par défaut. Pour plus d’informations sur les tables système, consultez Informations de référence sur les tables et les vues système.

Étape 1 : Créer la vue WLM_QUEUE_STATE_VW

Dans cette étape, vous créez une vue appelée WLM_QUEUE_STATE_VW. Cette vue retourne des informations à partir des tables système suivantes.

- STV_WLM_CLASSIFICATION_CONFIG

- STV_WLM_SERVICE_CLASS_CONFIG
- STV_WLM_SERVICE_CLASS_STATE

Vous utilisez cette vue tout au long du didacticiel pour surveiller ce qu'il advient des files d'attente, une fois que vous avez modifié la configuration de la gestion de la charge de travail. Le tableau suivante décrit les données que retourne la vue WLM_QUEUE_STATE_VW.

Colonne	Description
file d'attente	Numéro associé à la ligne qui représente une file d'attente. Le numéro de la file d'attente détermine l'ordre des files d'attente dans la base de données.
description	Valeur qui indique si la file d'attente est disponible uniquement pour certains groupes d'utilisateurs, pour certains groupes de requêtes ou pour tous les types de requêtes.
slots	Nombre d'emplacements alloués à la file d'attente.
mem	Quantité de mémoire allouée à la file d'attente, exprimée en Mo par emplacement.
max_execution_time	Durée pendant laquelle une requête est autorisée à s'exécuter avant d'être terminée.
user_*	Valeur qui indique si les caractères génériques sont autorisés dans la configuration WLM pour indiquer des groupes d'utilisateurs.
query_*	Valeur qui indique si les caractères génériques sont autorisés dans la configuration WLM pour indiquer des groupes de requêtes.
queued	Nombre de requêtes qui sont en attente dans la file d'attente pour être traitées.
executing	Nombre de requêtes qui sont en cours d'exécution.
executed	Nombre de requêtes exécutées.

Pour créer la vue WLM_QUEUE_STATE_VW

1. Ouvrez Amazon Redshift RSQL et connectez-vous à votre exemple de base de données TICKIT. Si vous n'avez pas cette base de données, consultez Prérequis.
2. Exécutez la requête suivante pour créer la vue WLM_QUEUE_STATE_VW.

```
create view WLM_QUEUE_STATE_VW as
select (config.service_class-5) as queue
, trim (class.condition) as description
, config.num_query_tasks as slots
, config.query_working_mem as mem
, config.max_execution_time as max_time
, config.user_group_wild_card as "user_*"
, config.query_group_wild_card as "query_*"
, state.num_queued_queries queued
, state.num_executing_queries executing
, state.num_executed_queries executed
from
STV_WLM_CLASSIFICATION_CONFIG class,
STV_WLM_SERVICE_CLASS_CONFIG config,
STV_WLM_SERVICE_CLASS_STATE state
where
class.action_service_class = config.service_class
and class.action_service_class = state.service_class
and config.service_class > 4
order by config.service_class;
```

3. Exécutez la requête suivante pour afficher les informations que la vue contient.

```
select * from wlm_queue_state_vw;
```

Voici un exemple de résultat.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (querytype:any)                           |     5 | 836 |        0 |  false |
 false   |      0 |         1 |      160
```

Étape 2 : Créer la vue WLM_QUEUE_STATE_VW

Dans cette étape, vous créez une vue appelée WLM_QUERY_STATE_VW. Cette vue retourne des informations à partir de la table système STV_WLM_QUERY_STATE.

Vous utilisez cette vue tout au long du didacticiel pour surveiller les requêtes en cours d'exécution. Le tableau suivant décrit les données que retourne la vue WLM_QUERY_STATE_VW.

Colonne	Description
query	ID de requête.
file d'attente	Numéro de la file d'attente.
slot_count	Nombre d'emplacements alloués à la requête.
start_time	Heure de début de la requête.
state	État de la requête (en cours d'exécution, par exemple).
queue_time	Nombre de microsecondes passées par la requête dans la file d'attente.
exec_time	Nombre de microsecondes pendant lesquelles la requête était en cours d'exécution.

Pour créer la vue WLM_QUERY_STATE_VW

1. Dans RSQL, exécutez la requête suivante pour créer la vue WLM_QUERY_STATE_VW.

```
create view WLM_QUERY_STATE_VW as
select query, (service_class-5) as queue, slot_count, trim(wlm_start_time) as
 start_time, trim(state) as state, trim(queue_time) as queue_time, trim(exec_time) as
 exec_time
from stv_wlm_query_state;
```

2. Exécutez la requête suivante pour afficher les informations que la vue contient.

```
select * from wlm_query_state_vw;
```

   Voici un exemple de résultat.

```
query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 1249 |     1 |          1 | 2014-09-24 22:19:16 | Executing | 0          | 516
```

Étape 3 : Exécuter les requêtes de test

Dans cette étape, vous exécutez des requêtes à partir de plusieurs connexions dans RSQL et examinez les tables système pour déterminer comment les requêtes ont été routées pour le traitement.

Pour cette étape, vous avez besoin d'avoir deux fenêtres RSQL ouvertes :

- Dans la première fenêtre RSQL, vous exécutez des requêtes qui surveillent l'état des files d'attente et des requêtes utilisant les vues que vous avez déjà créées dans ce didacticiel.
- Dans la deuxième fenêtre, vous exécutez des requêtes de longue durée pour modifier les résultats que vous trouvez dans la première fenêtre RSQL.

Pour exécuter les requêtes de test

1. Ouvrez deux fenêtres RSQL. Si vous avez déjà une fenêtre ouverte, il vous suffit d'en ouvrir une seconde. Vous pouvez choisir le même compte utilisateur pour ces deux connexions.
2. Dans la première fenêtre RSQL, exécutez la requête suivante.

   ```
   select * from wlm_query_state_vw;
   ```

   Voici un exemple de résultat.

   ```
   query | queue | slot_count | start_time          | state     | queue_time | exec_time
   ------+-------+------------+---------------------+-----------+------------
   +-----------
    1258 |     1 |          1 | 2014-09-24 22:21:03 | Executing | 0          | 549
   ```

   Cette requête retourne un résultat faisant référence à elle-même. La requête en cours d'exécution est l'instruction SELECT de cette vue. Une requête sur cette vue retourne toujours au moins un résultat. Comparez ce résultat à celui obtenu après le démarrage de la requête de longue durée à l'étape suivante.

3. Dans la deuxième fenêtre RSQL, exécutez une requête à partir de l'exemple de base de données TICKIT. Cette requête doit s'exécuter pendant une minute environ de telle sorte que vous ayez le temps d'explorer les résultats de la vue WLM_QUEUE_STATE_VW et de la vue WLM_QUERY_STATE_VW que vous avez créées précédemment. Dans certains cas, vous pouvez constater que la requête ne s'exécute pas assez longtemps pour que vous puissiez interroger les deux vues. Dans ces cas, vous pouvez augmenter la valeur du filtre sur `l.listid` pour qu'elle s'exécute plus longtemps.

   > **Note**
   >
   > Pour raccourcir le temps d'exécution des requêtes et améliorer les performances du système, Amazon Redshift met en cache le résultat de certains types de requêtes dans la mémoire du nœud principal. Lorsque la mise en cache des résultats est activée, les requêtes ultérieures s'exécutent beaucoup plus rapidement. Pour empêcher une exécution trop rapide de la requête, désactivez la mise en cache des résultats pour la séance en cours.

   Pour désactiver la mise en cache du résultat sur la séance en cours, définissez le paramètre enable_result_cache_for_session sur `off`, comme illustré ci-après.

   ```
   set enable_result_cache_for_session to off;
   ```

   Dans la deuxième fenêtre RSQL, exécutez la requête suivante.

   ```
   select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <
    100000;
   ```

4. Dans la première fenêtre RSQL, interrogez WLM_QUEUE_STATE_VW et WLM_QUERY_STATE_VW, puis comparez les résultats aux résultats précédents.

   ```
   select * from wlm_queue_state_vw;
   select * from wlm_query_state_vw;
   ```

   Voici quelques exemples de résultats.

   ```
   query | description                              | slots | mem | max_time | user_* |
    query_* | queued | executing | executed
   ```

```
------+----------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (querytype:any)                           |     5 | 836 |        0 |  false |
 false   |      0 |         2 |      163

query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 1267 |     1 |          1 | 2014-09-24 22:22:30 | Executing | 0          | 684
 1265 |     1 |          1 | 2014-09-24 22:22:36 | Executing | 0          | 4080859
```

Notez les différences suivantes entre vos requêtes précédentes et les résultats de cette étape :

- Il y a deux lignes maintenant dans WLM_QUERY_STATE_VW. Un résultat est la requête faisant référence à elle-même pour exécuter une opération SELECT sur cette vue. Le deuxième résultat est la longue requête de l'étape précédente.
- La colonne en cours d'exécution de WLM_QUEUE_STATE_VW a augmenté de 1 à 2. Cette entrée de colonne signifie qu'il y a deux requêtes en cours d'exécution dans la file d'attente.
- La colonne exécutée est augmentée chaque fois que vous exécutez une requête dans la file d'attente.

La vue WLM_QUEUE_STATE_VW est utile pour obtenir une vue d'ensemble des files d'attente et connaître le nombre de requêtes traitées dans chaque file d'attente. La vue WLM_QUERY_STATE_VW est utile pour obtenir une vue plus détaillée des requêtes individuelles en cours d'exécution.

Section 2 : Modification de la configuration de la file d'attente des requêtes WLM

Maintenant que vous connaissez le fonctionnement par défaut des files d'attente, vous pouvez apprendre à configurer les files d'attente de requêtes à l'aide de la gestion manuelle de la charge de travail. Dans cette section, vous créez et configurez un nouveau groupe de paramètres pour votre cluster. Vous créez deux files d'attente d'utilisateurs supplémentaires et les configurez pour accepter les requêtes basées sur le groupe d'utilisateurs des requêtes ou sur les étiquettes de groupe de requêtes. Toute requête qui n'est pas routée vers l'une de ces deux files d'attente est routée vers la file d'attente par défaut au moment de l'exécution.

Pour créer configuration WLM manuelle dans un groupe de paramètres

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Configurations, puis choisissez Gestion des charges de travail pour afficher la page Gestion des charges de travail.
3. Choisissez Créer pour afficher la fenêtre Créer un groupe de paramètres.
4. Saisissez **`WLMTutorial`** pour le Nom du groupe de paramètres et la Description, puis choisissez Créer pour créer le groupe de paramètres.

   > **Note**
   >
   > Le Nom du groupe de paramètres est converti au format minuscules pour toutes les lettres lors de sa création.

5. Sur la page Gestion des charges de travail, choisissez le groupe de paramètres **`wlmtutorial`** pour afficher la page des détails avec des onglets pour Paramètres et Gestion des charges de travail.
6. Vérifiez que vous êtes sur l'onglet Gestion des charges de travail, puis choisissez Passer en mode WLM pour afficher la fenêtre Paramètres de simultanéité.
7. Choisissez WLM manuel, puis Enregistrer pour passer à WLM manuel.
8. Choisissez Modifier les files d'attente de charge de travail.
9. Choisissez Ajouter une file d'attente deux fois pour ajouter deux files d'attente. Il y a désormais trois files d'attente : File d'attente 1, File d'attente 2 et File d'attente.
10. Entrez les informations de chaque file d'attente de la façon suivante :

    - Pour File d'attente 1, entrez **`30`** pour Mémoire (%), **`2`** pour Simultanéité sur principal et **`test`** pour Groupes de requêtes. Conservez les valeurs par défaut des autres paramètres.
    - Pour File d'attente 2, entrez **`40`** pour Mémoire (%), **`3`** pour Simultanéité sur principal et **`admin`** pour Groupes d'utilisateurs. Conservez les valeurs par défaut des autres paramètres.
    - Définissez la valeur Concurrency on main pour la file d'attente par défaut sur une valeur supérieure ou égale à 1. N'apportez aucune autre modification à la file d'attente par défaut. La gestion de la charge de travail affecte la mémoire non allouée à la file d'attente par défaut.

11. Choisissez Save (Enregistrer) pour enregistrer vos paramètres.

Ensuite, associez à un cluster le groupe de paramètres avec la configuration WLM manuelle.

Pour associer à un cluster un groupe de paramètres à une configuration WLM manuelle

1. Connectez-vous à la console Amazon Redshift AWS Management Console et ouvrez-la à l'adresse. https://console.aws.amazon.com/redshiftv2/
2. Dans le menu de navigation, choisissez Clusters, puis choisissez Clusters pour afficher la liste des clusters.
3. Choisissez votre cluster, par exemple `examplecluster`, pour afficher les informations du cluster. Sélectionnez ensuite l'option Propriétés pour afficher les propriétés de ce cluster.
4. Dans la section Configurations de base de données, sélectionnez Modifier, puis Modifier le groupe de paramètres pour afficher la fenêtre Groupes de paramètres.
5. Pour Groupes de paramètres, sélectionnez le groupe de paramètres **`wlmtutorial`** que vous avez précédemment créé.
6. Sélectionnez Enregistrer les modifications pour associer le groupe de paramètres.

   Le cluster est modifié avec le groupe de paramètres modifié. Toutefois, vous devez redémarrer le cluster pour que les modifications soient également appliquées à la base de données.
7. Choisissez votre cluster, puis choisissez Redémarrer le cluster comme Actions.

Une fois que le cluster a été redémarré, le statut redevient Disponible.

Section 3 : Acheminement des requêtes vers les files d'attente en fonction des groupes d'utilisateurs et des groupes de requêtes

À présent votre cluster est associé à un nouveau groupe de paramètres et vous avez configuré la gestion de la charge de travail. Ensuite, exécutez quelques requêtes pour voir comment Amazon Redshift achemine les requêtes dans les files d'attente pour le traitement.

Étape 1 : Afficher la configuration de la file d'attente de requêtes dans la base de données

D'abord, vérifiez que la base de données possède la configuration WLM que vous attendez.

Pour afficher la configuration de la file d'attente de requêtes

1. Ouvrez RSQL et exécutez la requête suivante. La requête utilise la vue WLM_QUEUE_STATE_VW que vous avez créée dans Étape 1 : Créer la vue

WLM_QUEUE_STATE_VW. Si vous aviez déjà une séance connectée à la base de données avant le redémarrage du cluster, vous devez vous reconnecter.

```
select * from wlm_queue_state_vw;
```

Voici un exemple de résultat.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         0 |        0
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         0 |        0
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |        0
```

Comparer ces résultats à ceux que vous avez reçus dans Étape 1 : Créer la vue WLM_QUEUE_STATE_VW. Notez qu'il y a désormais deux files d'attente supplémentaires. Queue 1 est maintenant la file d'attente du groupe de requêtes de test et Queue 2 la file d'attente du groupe utilisateur administrateur.

Queue 3 est maintenant la file d'attente par défaut. La dernière file d'attente de la liste est toujours la file d'attente par défaut. Elle correspond à la file d'attente vers laquelle les requêtes sont routées par défaut si aucun groupe d'utilisateurs ni groupe de requêtes n'est spécifié dans une requête.

2. Exécutez la requête suivante pour confirmer que votre requête s'exécute maintenant dans la file d'attente 3.

```
select * from wlm_query_state_vw;
```

Voici un exemple de résultat.

```
query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2144 |     3 |          1 | 2014-09-24 23:49:59 | Executing | 0          | 550430
```

Étape 2 : Exécuter une requête à l'aide de la file d'attente du groupe de requêtes

Pour exécuter une requête à l'aide de la file d'attente du groupe de requêtes

1. Exécutez la requête suivante pour l'acheminer vers le groupe de requêtes `test`.

```
set query_group to test;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

2. Dans l'autre fenêtre RSQL, exécutez la requête suivante.

```
select * from wlm_query_state_vw;
```

   Voici un exemple de résultat.

```
query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2168 |     1 |          1 | 2014-09-24 23:54:18 | Executing | 0          | 6343309
 2170 |     3 |          1 | 2014-09-24 23:54:24 | Executing | 0          | 847
```

   La requête a été acheminée vers le groupe de requêtes de test, à savoir queue 1 maintenant.

3. Sélectionnez tout à partir de la vue d'état de la file d'attente.

```
select * from wlm_queue_state_vw;
```

   Vous voyez un résultat similaire au suivant.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         1 |        0
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         0 |        0
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |        0
```

4. Maintenant, réinitialisez le groupe de requêtes et exécutez la longue requête à nouveau :

```
reset query_group;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

5. Exécutez les requêtes sur les vues pour afficher les résultats.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

   Voici quelques exemples de résultats.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         0 |        1
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         0 |        0
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         2 |        5

query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2186 |     3 |          1 | 2014-09-24 23:57:52 | Executing | 0          | 649
 2184 |     3 |          1 | 2014-09-24 23:57:48 | Executing | 0          | 4137349
```

   Le résultat doit être que la requête s'exécute maintenant dans la file d'attente 3 à nouveau.

Étape 3 : Créer un utilisateur de base de données et un groupe

Avant de pouvoir exécuter des requêtes dans cette file d'attente, vous devez créer le groupe d'utilisateurs dans la base de données et ajouter un utilisateur au groupe. Ensuite, vous vous connectez avec RSQL à l'aide des informations d'identification du nouvel utilisateur et exécutez les requêtes. Vous avez besoin d'exécuter des requêtes comme super-utilisateur, tel qu'utilisateur administrateur, pour pouvoir créer des utilisateurs de base de données.

Pour créer un utilisateur de base de données et un groupe d'utilisateurs

1. Dans la base de données, créez un utilisateur de base de données nommé `adminwlm` en exécutant la commande suivante dans une fenêtre RSQL.

```
create user adminwlm createuser password '123Admin';
```

2. Puis, exécutez les commandes suivantes pour créer le nouveau groupe d'utilisateurs et lui ajouter votre nouvel utilisateur `adminwlm`.

```
create group admin;
alter group admin add user adminwlm;
```

Étape 4 : Exécuter une requête à l'aide de la file d'attente du groupe d'utilisateurs

Ensuite, vous exécutez une requête et la routez vers la file d'attente du groupe d'utilisateurs. Vous procédez ainsi lorsque vous souhaitez acheminer votre requête vers une file d'attente configurée pour gérer le type de requête que vous souhaitez exécuter.

Pour exécuter une requête à l'aide de la file d'attente du groupe d'utilisateurs

1. Dans la deuxième fenêtre RSQL, exécutez les requêtes suivantes pour basculer vers le compte `adminwlm` et exécuter une requête en tant qu'utilisateur.

```
set session authorization 'adminwlm';
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

2. Dans la première fenêtre RSQL, exécutez la requête suivante pour voir vers quelle file d'attente les requêtes sont acheminées.

```
select * from wlm_query_state_vw;
select * from wlm_queue_state_vw;
```

   Voici quelques exemples de résultats.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
```

```
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         0 |        1
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         1 |        0
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |        8

query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2202 |     2 |          1 | 2014-09-25 00:01:38 | Executing | 0          | 4885796
 2204 |     3 |          1 | 2014-09-25 00:01:43 | Executing | 0          | 650
```

La file d'attente dans laquelle cette requête a été exécutée est la file d'attente 2, celle de l'utilisateur `admin`. Chaque fois que vous exécutez des requêtes en étant connecté sous l'identité de cet utilisateur, elles s'exécutent dans la file d'attente 2, sauf si vous spécifiez un autre groupe de requêtes à utiliser. La file d'attente choisie dépend des règles d'affectation de file d'attente. Pour de plus amples informations, veuillez consulter Règles d’affectation de file d’attente de WLM.

3. Maintenant, exécutez la requête suivante à partir de la deuxième fenêtre RSQL.

```
set query_group to test;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

4. Dans la première fenêtre RSQL, exécutez la requête suivante pour voir vers quelle file d'attente les requêtes sont acheminées.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

Voici quelques exemples de résultats.

```
query | description                               | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+-------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         1 |        1
```

```
    2 | (suser group: admin)                        |     3 | 557 |        0 |  false |
 false   |      0 |         0 |        1
    3 | (querytype:any)                             |     5 | 250 |        0 |  false |
 false   |      0 |         1 |       10

query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2218 |     1 |          1 | 2014-09-25 00:04:30 | Executing | 0          | 4819666
 2220 |     3 |          1 | 2014-09-25 00:04:35 | Executing | 0          | 685
```

5. Lorsque vous avez terminé, réinitialisez le groupe de requêtes.

```
reset query_group;
```

Section 4 : Utilisation de wlm_query_slot_count pour remplacer temporairement le niveau de concurrence d'une file d'attente

Parfois, les utilisateurs peuvent avoir temporairement besoin de plus de ressources pour une requête donnée. Si tel est le cas, ils peuvent utiliser le paramètre de configuration wlm_query_slot_count pour remplacer temporairement la manière dont les emplacements sont alloués dans une file d'attente de requêtes. Les emplacements correspondent aux unités de mémoire et à l'UC utilisées pour traiter les requêtes. Vous pouvez remplacer le nombre d'emplacements lorsque vous avez des requêtes occasionnelles qui nécessitent un grand nombre de ressources dans le cluster, comme lorsque vous effectuez une opération VACUUM dans la base de données.

Vous constatez peut-être que les utilisateurs ont souvent besoin de définir wlm_query_slot_count pour certains types de requêtes. Si tel est le cas, envisagez d'ajuster la configuration de la gestion de la charge de travail et de proposer aux utilisateurs une file d'attente mieux adaptée aux besoins de leurs requêtes. Pour de plus amples informations sur la substitution temporaire du niveau de concurrence à l'aide du nombre d'emplacements, veuillez consultez wlm_query_slot_count.

Étape 1 : Remplacer le niveau de concurrence à l'aide de wlm_query_slot_count

Dans le cadre de ce didacticiel, nous exécutons la même requête SELECT de longue durée. Nous l'exécutons sous l'identité de l'utilisateur `adminwlm` en utilisant wlm_query_slot_count pour augmenter le nombre d'emplacements disponibles pour la requête.

Pour remplacer le niveau de concurrence à l'aide de wlm_query_slot_count

1. Augmentez la limite de la requête pour vous assurer que vous avez assez de temps pour interroger la vue WLM_QUERY_STATE_VW et afficher un résultat.

```
set wlm_query_slot_count to 3;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

2. Maintenant, interrogez WLM_QUERY_STATE_VW avec l'utilisateur administrateur pour voir comment la requête s'exécute.

```
select * from wlm_query_state_vw;
```

   Voici un exemple de résultat.

```
query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2240 |     2 |          1 | 2014-09-25 00:08:45 | Executing | 0          | 3731414
 2242 |     3 |          1 | 2014-09-25 00:08:49 | Executing | 0          | 596
```

   Notez que le nombre d'emplacements pour la requête est 3. Ce nombre signifie que la requête utilise les trois emplacements pour traiter la requête et alloue la totalité des ressources de la file d'attente à cette requête.

3. Maintenant, exécutez la requête suivante.

```
select * from WLM_QUEUE_STATE_VW;
```

   Voici un exemple de résultat.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         0 |        4
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         1 |        3
```

```
    3 | (querytype:any)                          |     5 | 250 |        0 |  false |
 false   |      0 |         1 |       25
```

Le paramètre de configuration wlm_query_slot_count est valide pour la séance en cours uniquement. Si cette séance expire ou qu'un autre utilisateur exécute une requête, la configuration WLM est utilisée.

4. Réinitialisez le nombre d'emplacements et exécutez à nouveau le test.

```
reset wlm_query_slot_count;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

Voici quelques exemples de résultats.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      0 |         0 |        2
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         1 |        2
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |       14

query | queue | slot_count | start_time          | state     | queue_time | exec_time
------+-------+------------+---------------------+-----------+------------
+-----------
 2260 |     2 |          1 | 2014-09-25 00:12:11 | Executing | 0          | 4042618
 2262 |     3 |          1 | 2014-09-25 00:12:15 | Executing | 0          | 680
```

Étape 2 : Exécuter les requêtes à partir de différentes séances

Ensuite, exécutez les requêtes à partir de différentes séances.

Pour exécuter les requêtes à partir de différentes séances

1. Dans les première et deuxième fenêtres RSQL, exécutez la commande suivante pour utiliser le groupe de requêtes de test.

```
set query_group to test;
```

2. Dans la première fenêtre RSQL, exécutez la longue requête suivante.

```
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

3. Pendant que la requête de longue durée s'exécute dans la première fenêtre RSQL, exécutez les commandes suivantes. Elles augmentent le nombre d'emplacements pour utiliser tous les emplacements pour la file d'attente, puis commencent à exécuter la requête de longue durée.

```
set wlm_query_slot_count to 2;
select avg(l.priceperticket*s.qtysold) from listing l, sales s where l.listid <40000;
```

4. Ouvrez une troisième fenêtre RSQL et interrogez les vues pour afficher les résultats.

```
select * from wlm_queue_state_vw;
select * from wlm_query_state_vw;
```

Voici quelques exemples de résultats.

```
query | description                                | slots | mem | max_time | user_* |
 query_* | queued | executing | executed
------+--------------------------------------------+-------+-----+----------+--------
+---------+--------+-----------+----------
    0 | (super user) and (query group: superuser) |     1 | 357 |        0 |  false |
 false   |      0 |         0 |        0
    1 | (query group: test)                       |     2 | 627 |        0 |  false |
 false   |      1 |         1 |        2
    2 | (suser group: admin)                      |     3 | 557 |        0 |  false |
 false   |      0 |         0 |        3
    3 | (querytype:any)                           |     5 | 250 |        0 |  false |
 false   |      0 |         1 |       18

query | queue | slot_count | start_time          | state         | queue_time |
 exec_time
------+-------+------------+---------------------+---------------+------------
+-----------
 2286 |     1 |          2 | 2014-09-25 00:16:48 | QueuedWaiting | 3758950    | 0
 2282 |     1 |          1 | 2014-09-25 00:16:33 | Executing     | 0          |
 19335850
```

```
2288 |     3 |          1 | 2014-09-25 00:16:52 | Executing     | 0          | 666
```

Notez que la première requête utilise l'un des emplacements alloués à la file d'attente 1 pour exécuter la requête. De plus, notez qu'une requête est en attente dans la file d'attente (`queued` a pour valeur `1` et `state` a pour valeur `QueuedWaiting`). Une fois la première requête terminée, la deuxième commence à s'exécuter. Cette exécution se produit parce que les deux requêtes sont acheminées vers le groupe de requêtes `test` et la deuxième requête doit attendre qu'il y ait un nombre suffisant d'emplacements pour commencer le traitement.

Section 5 : Nettoyage de vos ressources

Votre cluster continue d'accumuler les frais aussi longtemps qu'il est en cours d'exécution. Lorsque vous avez terminé ce tutoriel, remettez votre environnement dans l'état précédent en suivant les étapes des rubriques Trouver des ressources supplémentaires et Réinitialiser votre environnement du Guide de démarrage d'Amazon Redshift.

Pour de plus amples informations sur la gestion de la charge de travail, veuillez consulter Gestion de la charge de travail.

Mise à l'échelle de la simultanéité

Grâce à la fonction de mise à l'échelle de la simultanéité, vous pouvez prendre en charge des milliers d'utilisateurs et de requêtes simultanées, avec des performances de requêtes toujours rapides. Lorsque vous activez la mise à l'échelle de la simultanéité, Amazon Redshift ajoute automatiquement de la capacité supplémentaire au cluster pour traiter un plus grand nombre de requêtes de lecture et d'écriture. Les utilisateurs voient les données les plus récentes, que les requêtes s'exécutent sur le cluster principal ou sur un cluster de mise à l'échelle de la simultanéité.

Vous pouvez gérer quelles requêtes sont envoyées au cluster de mise à l'échelle de la simultanéité en configurant les files d'attente WML. Lorsque vous activez la mise à l'échelle de la simultanéité, les requêtes éligibles sont envoyées au cluster de mise à l'échelle de la simultanéité au lieu d'attendre dans une file.

Vous êtes facturé pour les clusters de mise à l'échelle de la simultanéité uniquement pour la durée de leur utilisation. Pour plus d'informations sur les tarifs, y compris la façon dont les frais sont calculés et les frais minimaux, consultez Tarification de la mise à l'échelle de la simultanéité.

Rubriques

- Capacités de mise à l’échelle de la simultanéité
- Limitations de la mise à l’échelle de la simultanéité
- Régions AWS pour la mise à l'échelle simultanée
- Candidats à la mise à l’échelle de la simultanéité
- Configuration de files d’attente de mise à l’échelle de la simultanéité
- Surveillance de la mise à l’échelle de la simultanéité
- Vues système de la mise à l’échelle de la simultanéité

Capacités de mise à l’échelle de la simultanéité

Lorsque vous activez la mise à l’échelle de la simultanéité pour une file d’attente WLM, cela fonctionne pour les opérations de lecture, telles que les requêtes de tableau de bord. Cela fonctionne également pour les opérations d’écriture couramment utilisées, telles que les instructions pour l’ingestion et le traitement des données.

Capacités de mise à l’échelle de la simultanéité pour les opérations d’écriture

La mise à l’échelle de la simultanéité prend en charge les opérations d’écriture fréquemment utilisées, telles que les instructions Extract-transform-load (ETL). La mise à l’échelle de la simultanéité pour les opérations d’écriture est particulièrement utile lorsque vous souhaitez maintenir la cohérence des temps de réponse lorsque votre cluster reçoit un grand nombre de demandes. Cela améliore le débit pour les opérations d’écriture qui se disputent des ressources sur le cluster principal.

La mise à l’échelle de la simultanéité prend en charge les instructions COPY, INSERT, DELETE, UPDATE et CREATE TABLE AS (CTAS). En outre, le dimensionnement simultané prend en charge l'actualisation manuelle des vues matérialisées ()MVs. Les autres instructions DML (Data-Manipulation Language) et DDL (Data-Definition Language) ne sont pas prises en charge. Lorsque des instructions d’écriture non prises en charge, telles que CREATE sans TABLE AS, sont incluses dans une transaction explicite avant les instructions d’écriture prises en charge, aucune des instructions d’écriture n’est exécutée sur les clusters de mise à l’échelle de la simultanéité.

Lorsque vous accumulez des crédits pour la mise à l’échelle de la simultanéité, cette provision de crédits s’applique à la fois aux opérations de lecture et d’écriture.

Limitations de la mise à l'échelle de la simultanéité

Les restrictions suivantes s'appliquent à l'utilisation de l'adaptation de la simultanéité Amazon Redshift :

- Elle ne prend pas en charge les requêtes sur les tables qui utilisent les clés de tri entrelacées.
- Elle ne prend pas en charge les requêtes sur les tables temporaires.
- Elle ne prend pas en charge les requêtes qui accèdent aux ressources externes protégées par des configurations restrictives de réseau ou de cloud privé virtuel (VPC).
- Il ne prend pas en charge les requêtes contenant des fonctions Python définies par l'utilisateur (UDFs) et Lambda UDFs.
- Elle ne prend pas en charge les requêtes qui accèdent aux tables système, de catalogue PostgreSQL ou sans sauvegarde.
- Il ne prend pas en charge les requêtes COPY ou UNLOAD qui accèdent à une ressource externe lorsque des autorisations de politique IAM restrictives sont en place. Cela inclut les autorisations appliquées soit à la ressource, comme un bucket Amazon S3 ou une table DynamoDB, soit à la source. Les sources IAM peuvent inclure les suivantes :
  - `aws:sourceVpc`— Un VPC source.
  - `aws:sourceVpce`— Un point de terminaison VPC source.
  - `aws:sourceIp`— Une adresse IP source.

  Dans certains cas, vous devrez peut-être supprimer les autorisations qui limitent la ressource ou la source, afin que les requêtes COPY et UNLOAD accédant à la ressource soient envoyées au cluster de dimensionnement simultané.

  Pour plus d'informations sur les politiques de ressources, consultez les sections Types de politiques dans le guide de l' AWS Identity and Access Management utilisateur et Contrôle de l'accès depuis les points de terminaison VPC à l'aide de politiques de compartiment.
- L'adaptation de la simultanéité Amazon Redshift pour les opérations d'écriture n'est pas prise en charge pour les opérations DDL, telles que CREATE TABLE ou ALTER TABLE.
- Elle ne prend pas en charge ANALYZE pour la commande COPY.
- Elle ne prend pas en charge les opérations d'écriture sur une table cible où DISTSTYLE est défini sur ALL.
- Elle ne prend pas en charge la commande COPY à partir des formats de fichiers suivants :
  - Parquet

  - ORC
- Elle ne prend pas en charge les opérations d’écriture sur les tables avec des colonnes d’identité.
- Amazon Redshift prend en charge le dimensionnement simultané pour les opérations d'écriture uniquement sur les nœuds Amazon Redshift. RA3 La mise à l’échelle de la simultanéité pour les opérations d’écriture n’est pas prise en charge sur les autres types de nœuds.

Régions AWS pour la mise à l'échelle simultanée

Avec Amazon Redshift, vous pouvez utiliser le dimensionnement simultané pour gérer les demandes de charge de travail simultanées sur les clusters Redshift. Cette rubrique détaille les régions dans lesquelles vous pouvez utiliser le dimensionnement simultané avec Amazon Redshift.

La mise à l'échelle de la simultanéité est disponible dans les AWS régions suivantes :

- Région USA Est (Virginie du Nord) (us-east-1)
- Région USA Est (Ohio) (us-east-2)
- Région US Ouest (Californie du Nord) (us-west-1)
- Région USA Ouest (Oregon) (us-west-2)
- Région Asie-Pacifique (Mumbai) (ap-south-1)
- Région Asie-Pacifique (Séoul) (ap-northeast-2)
- Région Asie-Pacifique (Singapour) (ap-southeast-1)
- Région Asie-Pacifique (Sydney) (ap-southeast-2)
- Région Asie-Pacifique (Malaisie) (ap-southeast-5)
- Région Asie-Pacifique (Tokyo) (ap-northeast-1)
- Région Canada (Centre) (ca-central-1)
- Région Canada Ouest (Calgary) (ca-west-1)
- Région Chine (Beijing) (cn-north-1)
- Région Chine (Ningxia) (cn-northwest-1)
- Région Europe (Francfort) (eu-central-1)
- Région Europe (Irlande) (eu-west-1)
- Région Europe (Londres) (eu-west-2)
- Région Europe (Paris) (eu-west-3)

- Région Europe (Stockholm) (eu-north-1)
- Région Europe (Zurich) (eu-central-2)
- Région Europe (Espagne) (eu-south-2)
- Région Israël (Tel Aviv) (il-central-1)
- Région Amérique du Sud (Sao Paulo) (sa-east-1)
- AWS GovCloud (USA Est)

Candidats à la mise à l’échelle de la simultanéité

Avec Amazon Redshift, vous pouvez étendre le traitement des requêtes afin d'accélérer l'exécution de requêtes simultanées. La rubrique suivante décrit les critères utilisés par Amazon Redshift pour déterminer les requêtes à acheminer vers le dimensionnement de la simultanéité.

Les requêtes sont acheminées vers le cluster de mise à l’échelle de la simultanéité uniquement lorsque le cluster principal répond aux exigences suivantes :

- EC2-Plateforme VPC.
- Le type de nœud doit être dc2.8xlarge, dc2.large, ra3.large, ra3.xlplus, ra3.4xlarge ou ra3.16xlarge. Le dimensionnement simultané pour les opérations d'écriture est pris en charge uniquement sur les nœuds Amazon RA3 Redshift.
- 32 nœuds de calcul au maximum pour les clusters dotés de types de nœuds ra3.xlplus, ra3.4xlarge ou ra3.16xlarge. En outre, le nombre de nœuds du cluster principal ne peut pas dépasser 32 nœuds lorsque le cluster a été créé à l’origine. Par exemple, même si un cluster a actuellement 20 nœuds, mais qu’il a été créé à l’origine avec 40, il ne répond pas aux conditions requises pour la mise à l’échelle de la simultanéité. À l'inverse, si un DC2 cluster compte actuellement 40 nœuds, mais qu'il a été créé à l'origine avec 20 nœuds, il répond aux exigences relatives à la mise à l'échelle de la simultanéité.
- Ce n’est pas un cluster à nœud unique.

Configuration de files d’attente de mise à l’échelle de la simultanéité

Avec Amazon Redshift, vous pouvez gérer la simultanéité et les ressources système en configurant le dimensionnement de la simultanéité. Les files d'attente de dimensionnement simultané vous permettent de définir des limites quant au nombre de requêtes ou de sessions utilisateur pouvant être exécutées simultanément. La section suivante fournit des instructions sur la façon d'activer les

files d'attente de dimensionnement simultané dans Amazon Redshift, ce qui vous permet de gérer efficacement les requêtes simultanées et les sessions utilisateur.

Vous acheminez les requêtes vers des clusters de dimensionnement simultané en activant le dimensionnement de la simultanéité dans une file d'attente du gestionnaire de charge de travail (WLM). Pour activer la mise à l'échelle de la simultanéité sur une file d'attente, définissez la valeur de Concurrency Scaling mode sur auto.

Lorsque le nombre de requêtes acheminées vers une file d'attente pour laquelle le dimensionnement de la simultanéité est activé dépasse la capacité de simultanéité de la file d'attente, que la capacité soit configurée manuellement ou déterminée automatiquement, les requêtes éligibles sont envoyées au cluster de dimensionnement de la simultanéité. Lorsque des emplacements de file d'attente sont disponibles sur le cluster principal, les requêtes sont acheminées vers le cluster principal et exécutées sur celui-ci. Comme pour toute file d'attente WLM, vous acheminez les requêtes vers une file d'attente de dimensionnement de la simultanéité en fonction des groupes d'utilisateurs, en étiquetant les requêtes avec des étiquettes de groupes de requêtes ou en fonction des conditions de correspondance définies dans Affectation de requêtes aux files d'attente. Vous pouvez également acheminer les requêtes en définissant des Règles de surveillance de requête WLM. Par exemple, vous pouvez acheminer toutes les requêtes qui nécessitent plus de 5 secondes vers une file d'attente de mise à l'échelle de la simultanéité. N'oubliez pas que le comportement de mise en file d'attente peut varier selon que vous utilisez le WLM automatique ou le WLM manuel. Pour plus d'informations, consultez Implémentation du WLM automatique ou Implémentation du WLM manuel.

Par défaut, il y a un cluster pour la mise à l'échelle de la simultanéité. Le nombre de clusters de mise à l'échelle de la simultanéité qui peuvent être utilisés est contrôlé par max_concurrency_scaling_clusters.

Surveillance de la mise à l'échelle de la simultanéité

Avec Amazon Redshift, vous pouvez surveiller et gérer le dimensionnement simultané afin d'optimiser les performances et la rentabilité de vos charges de travail d'entreposage de données. Le dimensionnement simultané permet à Amazon Redshift d'ajouter automatiquement de la capacité de cluster supplémentaire lorsque la charge de travail augmente, et de supprimer cette capacité lorsque les demandes diminuent. La section suivante fournit des conseils sur la surveillance du dimensionnement de la simultanéité pour vos clusters Amazon Redshift.

Pour voir si une requête s'exécute sur le cluster principal ou sur un cluster de mise à l'échelle de la simultanéité, accédez à Cluster dans la console Amazon Redshift, puis choisissez un cluster.

Choisissez ensuite l'onglet Surveillance des requêtes et Simultanéité de charge de travail pour afficher des informations sur les requêtes en cours d'exécution et les requêtes en file d'attente.

Pour rechercher les temps d'exécution, interrogez la table STL_QUERY et filtrez sur la colonne `concurrency_scaling_status`. La requête suivante compare le temps d'attente et le temps d'exécution pour les requêtes exécutées sur le cluster de mise à l'échelle de la simultanéité et les requêtes exécutées sur le cluster principal.

```
SELECT w.service_class AS queue
, CASE WHEN q.concurrency_scaling_status = 1 THEN 'concurrency scaling cluster' ELSE
 'main cluster' END as concurrency_scaling_status
, COUNT( * ) AS queries
, SUM( q.aborted ) AS aborted
, SUM( ROUND( total_queue_time::NUMERIC / 1000000,2) ) AS queue_secs
, SUM( ROUND( total_exec_time::NUMERIC / 1000000,2) ) AS exec_secs
FROM stl_query q
JOIN stl_wlm_query w
USING (userid,query)
WHERE q.userid > 1
AND q.starttime > '2019-01-04 16:38:00'
AND q.endtime < '2019-01-04 17:40:00'
GROUP BY 1,2
ORDER BY 1,2;
```

Ajustez les valeurs `starttime` et `endtime` en fonction de vos besoins.

Vues système de la mise à l'échelle de la simultanéité

Avec Amazon Redshift, vous pouvez utiliser les vues du système de dimensionnement de la simultanéité pour surveiller et gérer les activités de dimensionnement de la simultanéité dans votre cluster. La section suivante décrit l'interrogation de ces vues système et l'interprétation des résultats afin de tirer efficacement parti du dimensionnement simultané dans votre environnement Amazon Redshift.

Un ensemble de vues système avec le préfixe SVCS fournit des détails à partir des tables du journal système concernant les requêtes sur le cluster principal et le cluster de mise à l'échelle de la simultanéité.

Les vues suivantes contiennent des informations similaires à celles des tables STL et des vues SVL correspondantes :

- SVCS_ALERT_EVENT_LOG
- SVCS_COMPILE
- SVCS_EXPLAIN
- SVCS_PLAN_INFO
- SVCS_QUERY_SUMMARY
- SVCS_STREAM_SEGS

Les vues suivantes sont spécifiques à la mise à l'échelle de la simultanéité :

- SVCS_CONCURRENCY_SCALING_USAGE

Pour plus d'informations sur la mise à l'échelle de la simultanéité, consultez les rubriques suivantes dans le Guide de la gestion du cluster Amazon Redshift.

- Affichage des données de mise à l'échelle de la simultanéité
- Affichage des performances de cluster pendant l'exécution des requêtes
- Affichage des détails de la requête

Accélération des requêtes courtes

L'accélération des requêtes courtes (SQA) établit la priorité des requêtes de courte durée sélectionnées sur les requêtes de longue durée. SQA exécute des requêtes de courte durée dans un espace dédié, afin que les requêtes SQA ne soient pas forcées d'attendre dans des files d'attente derrière les requêtes de longue durée. SQA priorise uniquement les requêtes de courte durée qui sont placées dans une file d'attente définie par l'utilisateur. Avec SQA, les requêtes de courte durée commencent à s'exécuter plus rapidement et les utilisateurs obtiennent les résultats plus rapidement.

Si vous activez SQA, vous pouvez réduire les files d'attente WLM qui sont dédiées à l'exécution des requêtes courtes. De plus, les requêtes de longue durée n'ont pas besoin de se heurter à des requêtes courtes pour obtenir des emplacements dans une file d'attente afin que vous puissiez configurer vos files d'attente WLM pour utiliser moins d'emplacements de requêtes. Lorsque vous utilisez une simultanéité réduite, le débit de requêtes est accru et les performances systèmes globales sont améliorées pour la plupart des charges de travail.

Les instructions CREATE TABLE AS (CTAS) et les requêtes en lecture seule, telles que les instructions SELECT, sont admissibles à la SQA.

Amazon Redshift utilise un algorithme de machine learning pour analyser chaque requête admissible et prédire l'heure d'exécution de la requête. Par défaut, WLM attribue dynamiquement une valeur à l'exécution maximale SQA en fonction de l'analyse de la charge de travail de votre cluster. Vous pouvez également spécifier une valeur fixe comprise entre 1 et 20 secondes. Si la durée d'exécution prévue de la requête est inférieure à la durée d'exécution maximale définie ou attribuée dynamiquement par la SQA et que la requête est en attente dans une file d'attente WLM, SQA sépare la requête des files d'attente WLM et la planifie pour une exécution prioritaire. Si une requête s'exécute en un délai supérieur au délai d'exécution maximal de la SQA, la gestion de la charge de travail déplace la requête vers la première file d'attente WLM correspondante en fonction des règles d'affectation de file d'attente de WLM. Au fil du temps, les prédiction s'améliorent à mesure que la SQA apprend de vos modèles de requêtes.

La SQA est activée par défaut dans le groupe de paramètres par défaut et pour tous les nouveaux groupes de paramètres. Pour désactiver la SQA dans la console Amazon Redshift, modifiez la configuration WLM pour un groupe de paramètres et désélectionnez Enable short query acceleration (Activer l'accélération des requêtes courtes). Nous vous recommandons d'utiliser un nombre d'emplacements de requêtes WLM de 15 ou inférieur pour conserver des performances système globales optimales. Pour plus d'informations sur la modification des configurations WLM, consultez Configuration de la gestion de la charge de travail dans le Guide de la gestion du cluster Amazon Redshift.

Durée d'exécution maximale pour les requêtes courtes

Lorsque vous activez SQA, WLM définit par défaut le délai d'exécution maximal des requêtes courtes avec la valeur « dynamic ». Il est recommandé de conserver cette valeur pour la durée d'exécution maximale SQA. Vous pouvez remplacer la valeur par défaut en spécifiant une valeur fixe comprise entre 1 et 20 secondes.

Dans certains cas, vous pouvez envisager d'utiliser des valeurs différentes pour la durée maximale d'exécution SQA afin d'améliorer vos performances système. Dans de tels cas, analysez votre charge de travail pour trouver la durée d'exécution maximale de la plupart de vos courtes requêtes. La requête suivante renvoie la durée d'exécution maximale pour les requêtes au 70e percentile (environ).

```
select least(greatest(percentile_cont(0.7)
within group (order by total_exec_time / 1000000) + 2, 2), 20)
from stl_wlm_query
where userid >= 100
```

```
and final_state = 'Completed';
```

Après avoir identifié une valeur de délai d'exécution maximal correspondant à votre charge de travail, vous n'avez plus besoin de le modifier, sauf si votre charge de travail change de manière significative.

Surveillance de la SQA

Pour vérifier si la SQA est activée, exécutez l'une des requêtes suivantes. Si la requête renvoie une ligne, la SQA est activée.

```
select * from stv_wlm_service_class_config
where service_class = 14;
```

La requête suivante affiche le nombre de requêtes ayant parcouru chaque file d'attente de requête (classe de service). Elle affiche aussi la durée d'exécution moyenne, le nombre de requêtes avec un délai d'attente au 90e percentile et le délai d'attente moyen. Les requêtes utilisent la classe de service 14.

```
select final_state, service_class, count(*), avg(total_exec_time),
percentile_cont(0.9) within group (order by total_queue_time), avg(total_queue_time)
from stl_wlm_query where userid >= 100 group by 1,2 order by 2,1;
```

Pour savoir quelles requêtes ont été prélevées par la SQA et exécutées avec succès, exécutez la requête suivante.

```
select a.queue_start_time, a.total_exec_time, label, trim(querytxt)
from stl_wlm_query a, stl_query b
where a.query = b.query and a.service_class = 14 and a.final_state = 'Completed'
order by b.query desc limit 5;
```

Pour savoir quelles requêtes ont été choisies par SQA mais ont expiré, exécutez la requête suivante.

```
select a.queue_start_time, a.total_exec_time, label, trim(querytxt)
from stl_wlm_query a, stl_query b
where a.query = b.query and a.service_class = 14 and a.final_state = 'Evicted'
order by b.query desc limit 5;
```

Pour plus d'informations sur les requêtes expulsées et, plus généralement, sur les actions basées sur des règles qui peuvent être entreprises sur les requêtes, consultez Règles de surveillance de requête WLM.

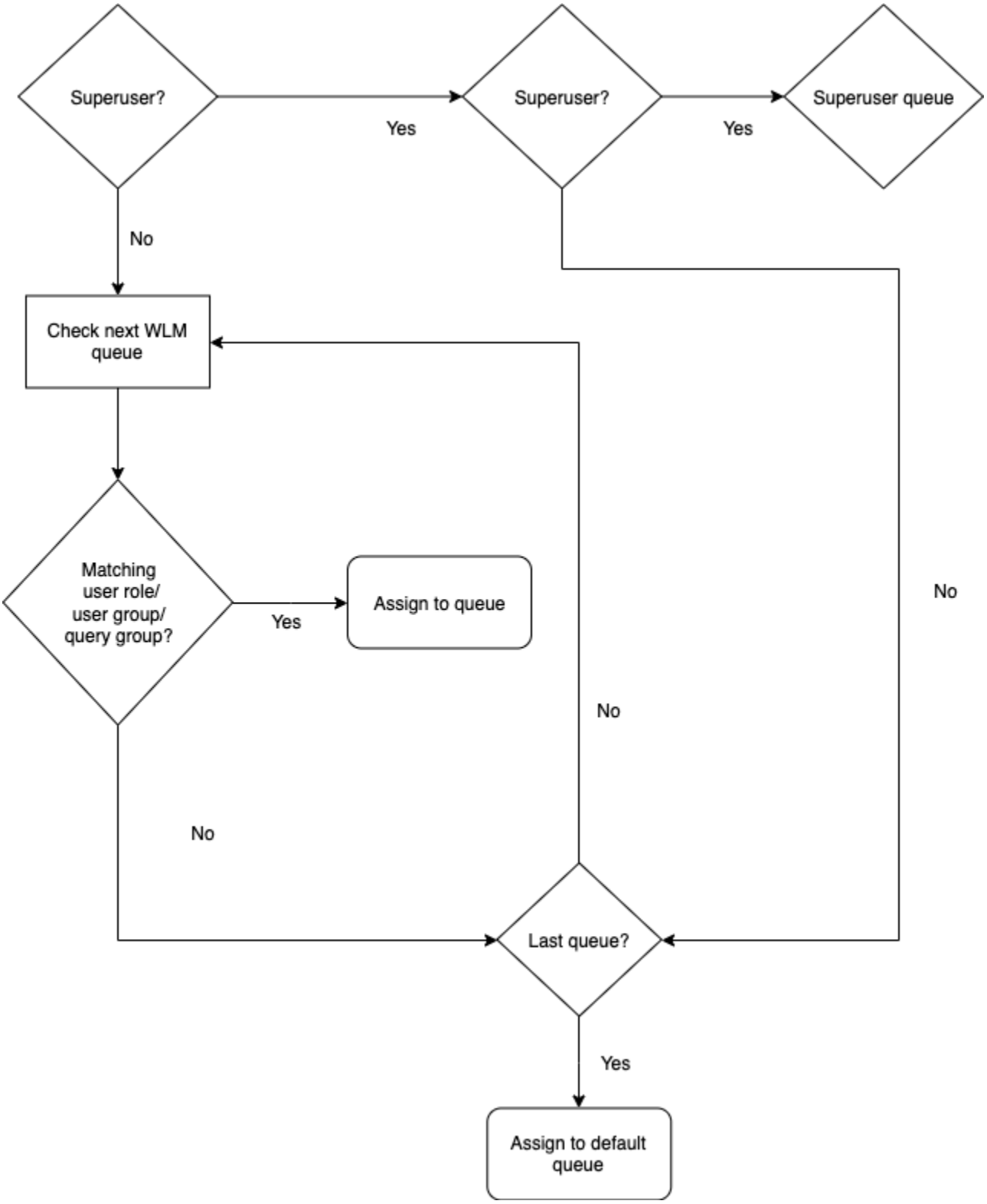
Règles d'affectation de file d'attente de WLM

Avec Amazon Redshift, vous pouvez contrôler l'allocation de mémoire et de ressources CPU aux requêtes des utilisateurs en définissant des règles d'attribution de files d'attente dans une configuration de gestion de charge de travail (WLM). La section suivante décrit la création et la gestion des règles d'attribution des files d'attente WLM afin d'allouer efficacement les ressources et de respecter les accords de niveau de service pour diverses charges de travail dans Amazon Redshift.

Lorsqu'un utilisateur exécute une requête, la gestion de la charge de travail affecte la requête à la première file d'attente correspondante, en fonction des règles d'affectation de file d'attente de la gestion de la charge de travail.

1. Si un utilisateur est connecté en tant que super-utilisateur et exécute une requête dans le groupe de requêtes étiqueté Super-utilisateur, la requête est affectée à la file d'attente Super-utilisateur.
2. Si un utilisateur fait partie d'un rôle, appartient à un groupe d'utilisateurs répertorié ou exécute une requête au sein d'un groupe de requêtes répertorié, celle-ci est affectée à la première file d'attente correspondante.
3. Si une requête ne respecte pas les critères, la requête est affectée à la file d'attente par défaut, qui est la dernière file d'attente définie dans la configuration WLM.

Le graphique suivant illustre le fonctionnement de ces règles.

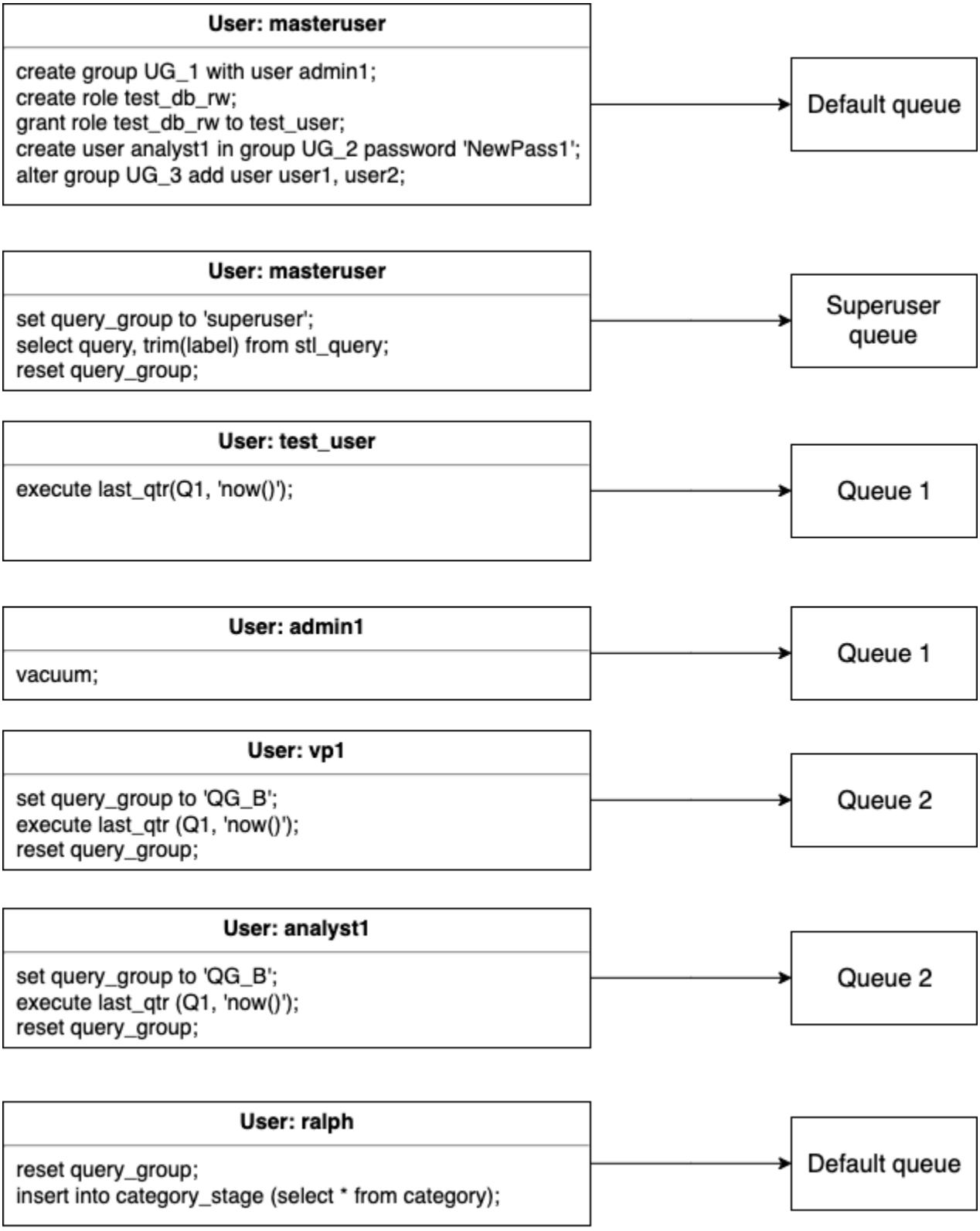

Superuser?
Yes
Superuser?
Yes
Superuser queue
No
Check next WLM queue
Matching user role/ user group/ query group?
Yes
Assign to queue
No
No
No
Last queue?
Yes
Assign to default queue

Exemple d’affectations de files d’attente

Le tableau suivant présente une configuration WLM avec la file d’attente Super-utilisateur et quatre files d’attente définies par l’utilisateur.

File d’attente	Simultanéité	Rôles utilisateur	Groupes d’utilisateurs	Groupes de requêtes
Superuser	1			super-utilisateur
1	5	test_db_rw	UG_1	
2	5			QG_B
3	5		UG_2	QG_C
Par défaut	5			

L’illustration suivante montre l’affectation des requêtes aux files d’attente de la table précédente en fonction des groupes d’utilisateurs et des groupes de requêtes. Pour plus d’informations sur l’affectation des requêtes à des groupes d’utilisateurs et à des groupes de requêtes lors de l’exécution, consultez Affectation des requêtes à des files d’attente ultérieurement dans cette section.



Dans cet exemple, WLM effectue les affectations suivantes :

1. La première série d'instructions présente trois façons d'affecter des utilisateurs à des groupes d'utilisateurs. Les instructions sont exécutées par l'utilisateur `adminuser`, qui n'est pas un membre d'un groupe d'utilisateurs répertorié dans une file d'attente WLM. Aucun groupe de requêtes n'est défini, les déclarations sont donc acheminées vers la file d'attente par défaut.
2. L'utilisateur `adminuser` est un super-utilisateur et le groupe de requêtes est défini sur `'superuser'`, la requête est donc affectée à la file d'attente du super-utilisateur.
3. L'utilisateur `test_user` se voit attribuer le rôle `test_db_rw` répertorié dans la file d'attente 1, la requête est donc affectée à la file d'attente 1.
4. L'utilisateur `admin1` est membre du groupe d'utilisateurs répertorié dans la file d'attente 1, la requête est donc affectée à la file d'attente 1.
5. L'utilisateur `vp1` n'est membre d'aucun groupe d'utilisateurs répertorié. Le groupe de requêtes est défini sur `'QG_B'`, la requête est donc affectée à la file d'attente 2.
6. L'utilisateur `analyst1` est membre du groupe d'utilisateurs répertorié dans la file d'attente 3, mais `'QG_B'` correspond à la file d'attente 2, la requête est donc affectée à la file d'attente 2.
7. L'utilisateur `ralph` n'est membre d'aucun groupe d'utilisateurs répertorié et le groupe de requêtes a été réinitialisé, il n'y a aucune donc file d'attente correspondante. La requête est affectée à la file d'attente par défaut.

Affectation des requêtes à des files d'attente

Avec Amazon Redshift, vous pouvez gérer la simultanéité des charges de travail et hiérarchiser les requêtes en les affectant à des files d'attente. Les files d'attente vous permettent d'allouer des ressources telles que la mémoire et le processeur à différents types de requêtes ou d'utilisateurs, garantissant ainsi la priorité des requêtes critiques par rapport aux requêtes moins importantes. Les sections suivantes décrivent comment créer des files d'attente, configurer leurs propriétés et attribuer les requêtes entrantes en fonction des critères que vous définissez.

Les exemples suivants affectent des requêtes à des files d'attente en fonction des rôles utilisateurs, des groupes d'utilisateurs et des groupes de requêtes.

Affectation des requêtes aux files d'attente en fonction des rôles utilisateurs

Si un utilisateur se voit attribuer un rôle associé à une file d'attente, les requêtes exécutées par cet utilisateur sont affectées à cette file d'attente. L'exemple suivant crée un rôle utilisateur nommé `sales_rw` et attribue ce rôle à l'utilisateur `test_user`.

```
create role sales_rw;
grant role sales_rw to test_user;
```

Vous pouvez également combiner les autorisations de deux rôles en attribuant explicitement un rôle à un autre rôle. L'attribution d'un rôle imbriqué à un utilisateur permet à l'utilisateur d'accéder aux deux rôles.

```
create role sales_rw;
create role sales_ro;
grant role sales_ro to role sales_rw;
grant role sales_rw to test_user;
```

Pour voir la liste des utilisateurs auxquels des rôles ont été attribués dans le cluster, consultez la table SVV_USER_GRANTS. Pour voir la liste des rôles auxquels des rôles ont été attribués dans le cluster, consultez la table SVV_ROLE_GRANTS.

```
select * from svv_user_grants;
select * from svv_role_grants;
```

Affectation des requêtes aux files d'attente en fonction des groupes d'utilisateurs

Si un nom de groupe d'utilisateurs est répertorié dans une définition de file d'attente, les requêtes exécutées par des membres de ce groupe d'utilisateurs sont affectées à la file d'attente correspondante. L'exemple suivant crée des groupes d'utilisateurs et ajoute des utilisateurs à des groupes à l'aide des commandes SQL CREATE USER, CREATE GROUP et ALTER GROUP.

```
create group admin_group with user admin246, admin135, sec555;
create user vp1234 in group ad_hoc_group password 'vpPass1234';
alter group admin_group add user analyst44, analyst45, analyst46;
```

Affectation d'une requête à un groupe de requêtes

Vous pouvez affecter une requête à une file d'attente lors de l'exécution en affectant votre requête au groupe de requêtes approprié. Utilisez la commande SET pour commencer un groupe de requêtes.

```
SET query_group TO group_label
```

Ici, *group_label* est une étiquette de groupe de requête qui est répertoriée dans la configuration de WLM.

Toutes les requêtes que vous exécutez après l'exécution de la commande SET query_group s'exécutent comme membres du groupe de requêtes spécifié jusqu'à ce que vous réinitialisiez le groupe de requêtes ou que vous mettiez fin à votre session de connexion actuelle. Pour plus d'informations sur la définition et la réinitialisation des objets Amazon Redshift, consultez SET et RESET dans le Guide de référence des commandes SQL.

Les étiquettes de groupe de requêtes que vous spécifiez doivent être incluses dans la configuration WLM actuelle, sinon la commande SET query_group n'aura aucune incidence sur les files d'attente de requêtes.

L'étiquette définie dans la clause TO est capturée dans les journaux de requêtes afin que vous puissiez utiliser l'étiquette à des fins de résolution de problèmes. Pour plus d'informations sur le paramètre de configuration query_group, consultez query_group dans la Référence de configuration.

L'exemple suivant exécute deux requêtes dans le cadre du groupe de requêtes 'priority', puis réinitialise le groupe de requêtes.

```
set query_group to 'priority';
select count(*)from stv_blocklist;
select query, elapsed, substring from svl_qlog order by query desc limit 5;
reset query_group;
```

Affectation des requêtes à la file d'attente Super-utilisateur

Pour affecter une requête à la file d'attente superutilisateur, connectez-vous à Amazon Redshift en tant que superutilisateur, puis exécutez la requête dans le groupe de celui-ci. Une fois terminé, réinitialisez le groupe de requêtes afin que les requêtes suivantes ne s'exécutent pas dans la file d'attente Super-utilisateur.

L'exemple suivant affecte deux commandes à exécuter dans la file d'attente Super-utilisateur.

```
set query_group to 'superuser';

analyze;
vacuum;
reset query_group;
```

Pour afficher la liste des super-utilisateurs, interrogez la table catalogue système PG_USER.

```
select * from pg_user where usesuper = 'true';
```

Propriétés de configuration dynamiques et statiques WLM

Les propriétés de la configuration WLM sont dynamiques ou statiques. Vous pouvez appliquer des propriétés dynamiques à la base de données sans redémarrage du cluster, mais les propriétés statiques nécessitent un redémarrage du cluster pour que les modifications prennent effet. Cependant, si vous modifiez les propriétés dynamiques et statiques en même temps, vous devez redémarrer le cluster pour que toutes les modifications prennent effet. Cela se vérifie que les propriétés soient statiques ou dynamiques.

Pendant l'application des propriétés dynamiques, votre cluster a le statut `modifying`. Le basculement entre le WLM automatique et le WLM manuel est une modification statique et nécessite une réinitialisation de cluster pour être pris en compte.

La table suivante indique si les propriétés WLM sont dynamiques ou statiques lors de l'utilisation de la gestion automatique ou manuelle de la charge de travail.

Propriété WLM	Gestion automatique de la charge de travail	Gestion manuelle de la charge de travail
Groupes de requêtes	Répartition dynamique	Statique
Caractère générique de groupe de requêtes	Répartition dynamique	Statique
Groupes d'utilisateurs	Répartition dynamique	Statique
Caractère générique de groupe d'utilisateurs	Répartition dynamique	Statique
Rôles utilisateurs	Répartition dynamique	Statique
Caractère générique du rôle utilisateur	Répartition dynamique	Statique

Propriété WLM	Gestion automatique de la charge de travail	Gestion manuelle de la charge de travail
Concurrency on main (Simultanéité sur cluster principal)	Ne s'applique pas	Répartition dynamique
Mode de mise à l'échelle de la simultanéité	Répartition dynamique	Répartition dynamique
Activer l'accélération des requêtes courtes	Ne s'applique pas	Répartition dynamique
Durée d'exécution maximale pour les requêtes courtes	Répartition dynamique	Répartition dynamique
Pourcentage de mémoire à utiliser	Ne s'applique pas	Répartition dynamique
Expiration	Ne s'applique pas	Répartition dynamique
Priorité	Répartition dynamique	Ne s'applique pas
Ajout ou suppression de files d'attente	Répartition dynamique	Statique

Si vous ajoutez une règle de surveillance des requêtes (QMR), ou si vous modifiez ou supprimez une QMR existante, le changement s'effectue automatiquement sans qu'il soit nécessaire de redémarrer le cluster.

> **Note**
>
> Lors de l'utilisation de la gestion manuelle de charge de travail, si la valeur du délai d'attente est modifiée, la nouvelle valeur est appliquée à toute requête dont l'exécution démarre une fois que la valeur a été modifiée. Si la simultanéité ou le pourcentage de mémoire à utiliser sont modifiés, Amazon Redshift passe à la nouvelle configuration de manière dynamique. Les requêtes en cours d'exécution ne sont donc pas affectées par le changement. Pour plus d'informations, consultez Allocation de mémoire dynamique WLM.

Rubriques

- Allocation de mémoire dynamique WLM
- Exemple WLM dynamique

Allocation de mémoire dynamique WLM

Dans chaque file d'attente, WLM crée un certain nombre d'emplacements de requêtes équivalant au niveau de simultanéité de la file d'attente. La quantité de mémoire allouée à un emplacement de requête équivaut au pourcentage de mémoire allouée à la file d'attente divisé par le nombre d'emplacements. Si vous modifiez l'allocation de mémoire ou la simultanéité, Amazon Redshift gère dynamiquement la transition vers la nouvelle configuration de WLM. Les requêtes actives peuvent donc s'exécuter jusqu'à ce qu'elles soient terminées en utilisant la quantité de mémoire actuellement allouée. Dans le même temps, Amazon Redshift veille à ce que l'utilisation totale de la mémoire ne dépasse jamais 100 % de la mémoire disponible.

Le gestionnaire de la charge de travail utilise le processus suivant pour gérer la transition.

1. WLM recalcule l'allocation de mémoire de chaque nouvel emplacement de requête.
2. Si un emplacement de requête n'est pas utilisé activement par une requête en cours d'exécution, WLM supprime l'emplacement, ce qui fournit de la mémoire disponible pour les nouveaux emplacements.
3. Si un emplacement de requête est activement en cours d'utilisation, WLM attend la fin de la requête.
4. A mesure que les requêtes actives prennent fin, les emplacements vides sont supprimés et la mémoire associée est libérée.
5. Lorsque suffisamment de mémoire est disponible pour ajouter un ou plusieurs emplacements, de nouveaux emplacements sont ajoutés.
6. Une fois que toutes les requêtes qui étaient en cours d'exécution au moment de la modification prennent fin, le nombre d'emplacements équivaut au nouveau niveau de simultanéité, et la transition vers la nouvelle configuration WLM est terminée.

Dans les faits, les requêtes en cours d'exécution au moment de la modification continuent d'utiliser l'allocation de mémoire d'origine. Les requêtes en file d'attente au moment de la modification sont dirigées vers de nouveaux emplacements, au fur et à mesure qu'ils deviennent disponibles.

Si les propriétés dynamiques WLM sont modifiées au cours du processus de transition, WLM commence immédiatement à passer à la nouvelle configuration, à partir de l'état actuel. Pour afficher le statut de la transition, interrogez la table système STV_WLM_SERVICE_CLASS_CONFIG.

Exemple WLM dynamique

Avec Amazon Redshift, vous pouvez gérer automatiquement la distribution de la charge de travail et l'allocation des ressources sur vos clusters Amazon Redshift à l'aide de Dynamic WLM (gestion de la charge de travail). Le WLM dynamique est un exemple de configuration de gestion de la charge de travail (WLM) qui ajuste dynamiquement les allocations de mémoire en fonction des demandes de charge de travail, permettant ainsi une simultanéité et des performances optimales. La section suivante fournit des informations sur la mise en œuvre et la configuration de Dynamic WLM pour vos clusters Amazon Redshift.

Supposons que votre cluster WLM soit configuré avec deux files d'attente, à l'aide des propriétés dynamiques suivantes.

File d'attente	Simultanéité	% de mémoire à utiliser
1	4	50%
2	4	50%

Supposons à présent que votre cluster dispose de 200 Go de mémoire disponible pour le traitement de la requête. (Ce nombre est arbitraire et utilisé à titre d'illustration uniquement.) Comme le montre l'équation suivante, chaque emplacement se voit allouer 25 Go.

```
(200 GB * 50% ) / 4 slots  = 25 GB
```

Ensuite, vous modifiez votre WLM pour utiliser les propriétés dynamiques suivantes.

File d'attente	Simultanéité	% de mémoire à utiliser
1	3	75%
2	4	25%

Comme le montre l'équation suivante, la nouvelle allocation de mémoire de chaque emplacement de la file d'attente 1 est de 50 Go.

```
(200 GB * 75% ) / 3 slots = 50 GB
```

Supposons que les requêtes A1, A2, A3 et A4 soient en cours d'exécution lorsque la nouvelle configuration est appliquée, et que les requêtes B1, B2, B3 et B4 sont mises en file d'attente. WLM reconfigure dynamiquement les emplacements de requêtes comme suit.

Étape	Requêtes en cours d'exécution	Nombre d'emplacements actuel	Nombre d'emplacements cible	Mémoire allouée	Mémoire disponible
1	A1, A2, A3, A4	4	0	100 Go	50 Go
2	A2, A3, A4	3	0	75 Go	75 Go
3	A3, A4	2	0	50 Go	100 Go
4	A3, A4, B1	2	1	100 Go	50 Go
5	A4, B1	1	1	75 Go	75 Go
6	A4, B1, B2	1	2	125 Go	25 Go
7	B1, B2	0	2	100 Go	50 Go
8	B1, B2, B3	0	3	150 Go	0 Go

1. WLM recalcule l'allocation de mémoire de chaque emplacement de requête. À l'origine, la file d'attente 1 s'est vue allouer 100 Go. La nouvelle file d'attente dispose d'une allocation totale de 150 Go, elle a donc immédiatement 50 Go disponibles. La file d'attente 1 utilise à présent quatre emplacements, et le nouveau niveau de simultanéité est de trois emplacements, aucun nouvel emplacement n'est donc ajouté.
2. Lorsqu'une requête se termine, l'emplacement est supprimé et 25 Go sont libérés. La file d'attente 1 compte à présent trois emplacements et 75 Go de mémoire disponible. La nouvelle

configuration nécessite 50 Go pour chaque nouvel emplacement, mais le nouveau niveau de simultanéité est de trois emplacements, aucun nouvel emplacement n'est donc ajouté.

3. Lorsqu'une deuxième requête se termine, l'emplacement est supprimé et 25 Go sont libérés. La file d'attente 1 compte à présent deux emplacements et 100 Go de mémoire disponible.
4. Un nouvel emplacement est ajouté à l'aide des 50 Go de mémoire disponible. La file d'attente 1 compte à présent trois emplacements et 50 Go de mémoire libre. Les requêtes mises en file d'attente peuvent désormais être acheminées vers le nouvel emplacement.
5. Lorsqu'une troisième requête se termine, l'emplacement est supprimé et 25 Go sont libérés. La file d'attente 1 compte à présent deux emplacements et 75 Go de mémoire disponible.
6. Un nouvel emplacement est ajouté à l'aide des 50 Go de mémoire disponible. La file d'attente 1 compte à présent trois emplacements et 25 Go de mémoire libre. Les requêtes mises en file d'attente peuvent désormais être acheminées vers le nouvel emplacement.
7. Lorsqu'une quatrième requête se termine, l'emplacement est supprimé et 25 Go sont libérés. La file d'attente 1 compte à présent deux emplacements et 50 Go de mémoire disponible.
8. Un nouvel emplacement est ajouté à l'aide des 50 Go de mémoire disponible. La file d'attente 1 compte à présent trois emplacements de 50 Go chacun et toute la mémoire disponible a été allouée.

La transition est terminée, et tous les emplacements de requêtes sont disponibles pour les requêtes mises en file d'attente.

Règles de surveillance de requête WLM

Dans la gestion des charges de travail (WLM) d'Amazon Redshift, les règles de surveillance des requêtes définissent des limites de performance basées sur des mesures pour les files d'attente WLM et spécifient les mesures à prendre lorsqu'une requête dépasse ces limites. Par exemple, pour une file d'attente dédiée aux requêtes de courte durée, vous pouvez créer une règle qui annule les requêtes qui s'exécutent pendant plus de 60 secondes. Si vous souhaitez effectuer un suivi des requêtes mal conçues, vous pouvez configurer une autre règle qui consigne les requêtes contenant des boucles imbriquées.

Vous définissez les règles de surveillance de requête dans le cadre de la configuration de la gestion de la charge de travail (WLM). Vous pouvez définir jusqu'à 25 règles par file d'attente, et jusqu'à 25 règles au total pour toutes les files d'attente. Chaque règle est composée au maximum de trois conditions (prédicats) et d'une action. Un prédicat est composé d'une métrique, d'une condition de

comparaison (=, < ou >) et d'une valeur. Si les conditions de l'ensemble des prédicats d'une règle sont respectées, l'action associée à cette règle est déclenchée. Les règles peuvent être associées aux actions log, hop et abort, comme discuté ci-après.

Les règles dans une file d'attente donnée s'appliquent uniquement aux requêtes en cours d'exécution dans cette file d'attente. Une règle est indépendante des autres règles.

WLM évalue les métriques toutes les 10 secondes. Amazon Redshift applique des règles de surveillance des requêtes au niveau de la requête enfant lorsque les requêtes sont automatiquement réécrites. Si plusieurs règles sont déclenchées au cours de la même période, WLM choisit la règle comportant l'action la plus sévère. Si l'action pour deux règles a la même sévérité, WLM exécute les règles par ordre alphabétique, en fonction du nom de la règle. Si l'action est hop ou abort, elle est consignée et la requête est expulsée de la file d'attente. Si l'action est log, la requête continue à s'exécuter dans la file d'attente. WLM déclenche une seule action log par requête et par règle. Si la file d'attente contient d'autres règles, celles-ci restent en vigueur. Si l'action est hop et que la requête est acheminée vers une autre file d'attente, les règles relatives à la nouvelle file d'attente s'appliquent. Pour plus d'informations sur la surveillance des requêtes et le suivi des actions entreprises sur des requêtes spécifiques, consultez la collection d'exemples sur Accélération des requêtes courtes.

Lorsque l'ensemble des prédicats d'une règle sont respectés, WLM écrit une ligne dans la table système STL_WLM_RULE_ACTION. En outre, Amazon Redshift enregistre les métriques des requêtes en cours d'exécution dans STV_QUERY_METRICS. Les métriques des requêtes terminées sont stockées dans STL_QUERY_METRICS.

Définition d'une règle de surveillance de requête

Vous créez des règles de surveillance de requête dans le cadre de la configuration WLM, que vous définissez lors de la définition du groupe de paramètres de votre cluster.

Vous pouvez créer des règles à l'aide du AWS Management Console JSON ou par programmation.

> **Note**
>
> Si vous créez des règles par programmation, il est recommandé d'utiliser la console afin de générer les données JSON à inclure dans la définition du groupe de paramètres. Pour plus d'informations, consultez Création ou modification d'une règle de surveillance de requête à l'aide de la console et Configuration des valeurs des paramètres à l'aide de l' AWS CLI dans le Guide de la gestion du cluster Amazon Redshift.

Pour définir une règle de surveillance de requête, spécifiez les éléments suivants :

- Un nom de règle – Les noms des règles doivent être uniques au sein de la configuration WLM. Les noms de règles peut comporter jusqu'à 32 caractères alphanumériques ou traits de soulignement et ne doivent pas contenir d'espaces ni de guillemets. Vous pouvez disposer de jusqu'à 25 règles par file d'attente et la limite totale pour toutes les files d'attente est de 25 règles.
- Un ou plusieurs prédicats – Chaque règle peut comporter jusqu'à trois prédicats. Si l'ensemble des prédicats d'une règle sont respectés, l'action associée est déclenchée. Un prédicat se définit par un nom de métrique, un opérateur (=, < ou >) et une valeur. Par exemple : `query_cpu_time > 100000`. Pour obtenir une liste des métriques, ainsi que des exemples de valeurs pour différentes métriques, voir Métriques de surveillance des requêtes pour cluster Amazon Redshift provisionné plus loin dans cette section.
- Une action – Si plusieurs règles sont déclenchées, WLM choisit celle dont l'action est la plus grave. Les actions possibles sont les suivantes, par ordre croissant de gravité :
  - Log – Cette action enregistre des informations sur la requête dans la table système STL_WLM_RULE_ACTION. Utilisez cette action si vous souhaitez uniquement écrire un enregistrement de journal. WLM crée au maximum un journal par requête et par règle. Suite à une action log, les autres règles restent en vigueur et WLM continue à surveiller la requête.
  - Hop (uniquement disponible avec la gestion manuelle de la charge de travail) – Consigne l'action et fait sauter la requête vers la file d'attente correspondante suivante. S'il n'existe aucune autre file d'attente correspondante, la requête est annulée. QMR replace uniquement les instructions CREATE TABLE AS (CTAS) et les requêtes en lecture seule, telles que les instructions SELECT. Pour de plus amples informations, veuillez consulter Saut de file d'attente des requêtes WLM.
  - Abort – Journalise l'action et annule la requête. QMR n'arrête pas les instructions COPY et les opérations de maintenance, telles que ALTER, ANALYZE et VACUUM.
  - Change priority (Modifier la priorité) (uniquement disponible avec la gestion automatique de la charge de travail) – Pour modifier la priorité d'une requête.

Pour limiter la durée d'exécution des requêtes, nous vous recommandons de créer une règle de surveillance de requête au lieu d'utiliser le délai WLM. Par exemple, vous pouvez définir `max_execution_time` sur 50 000 millisecondes, comme illustré dans cet extrait JSON.

```
"max_execution_time": 50000
```

Mais nous vous recommandons plutôt de définir une règle de surveillance des requêtes équivalente. L'exemple suivant illustre une règle de surveillance des requêtes définie `query_execution_time` sur 50 secondes :

```
"rules":
[
    {
        "rule_name": "rule_query_execution",
        "predicate": [
            {
                "metric_name": "query_execution_time",
                "operator": ">",
                "value": 50
            }
        ],
        "action": "abort"
    }
]
```

Pour les étapes de création ou de modification d'une règle de surveillance des requêtes, consultez Création ou modification d'une règle de surveillance des requêtes à l'aide de la console et Propriétés du paramètre wlm_json_configuration dans le Guide de la gestion du cluster Amazon Redshift.

Vous trouverez plus d'informations sur les règles de surveillance de requête dans les rubriques suivantes :

- Métriques de surveillance des requêtes pour cluster Amazon Redshift provisionné
- Modèles de règles de surveillance de requête
- Création d'une règle à l'aide de la console
- Configuration de la gestion de la charge de travail
- Tables et vues système pour les règles de surveillance de requête

Métriques de surveillance des requêtes pour cluster Amazon Redshift provisionné

Le tableau suivant décrit les métriques utilisées dans les règles de surveillance de requête. (Ces métriques diffèrent des métriques stockées dans les tables système STV_QUERY_METRICS et STL_QUERY_METRICS.)

Pour une métrique donnée, le seuil de performance fait l'objet d'un suivi au niveau de la requête ou du segment. Pour plus d'informations sur les segments et les étapes, consultezWorkflow d'exécution et de planification de requête.

> **Note**
>
> Le paramètre Délai WLM diffère des règles de surveillance de requête.

Métrique	Name (Nom)	Description
Temps UC de la requête	`query_cpu_time`	Temps UC utilisé par la requête (en secondes) . `CPU time` diffère de `Query execution time`. Les valeurs valides sont comprises entre 0 et 999 999.
Blocs lus	`query_blocks_read`	Nombre de blocs de données d'1 Mo lus par la requête. Les valeurs valides sont comprises entre 0 et 1 048 575.
Nombre de lignes d'analyse	`scan_row_count`	Nombre de lignes dans une étape d'analyse. Le nombre de lignes correspond au nombre total de lignes émises avant le filtrage des lignes marquées pour la suppression (lignes fantôme) et avant l'application des filtres de requête définis par l'utilisateur. Les valeurs valides sont comprises entre 0 et 999 999 999 999 999.
Durée d'exécution de la requête	`query_execution_time`	Délai écoulé pour l'exécution d'une requête (en secondes). Le délai d'exécution n'inclut pas le temps d'attente dans une file d'attente.

Métrique	Name (Nom)	Description
		Les valeurs valides sont comprises entre 0 et 86 399.
Temps de file d'attente de la requête	`query_queue_time`	Temps passé à attendre dans une file d'attente, en secondes. Les valeurs valides sont comprises entre 0 et 86 399.
Utilisation de l'UC	`query_cpu_usage_percent`	Pourcentage de la capacité de l'UC utilisée par la requête. Les valeurs valides sont comprises entre 0 et 6 399.
Mémoire sur disque	`query_temp_blocks_to_disk`	Espace disque temporairement utilisé pour écrire des résultats intermédiaires, en blocs d'1 Mo. Les valeurs valides sont comprises entre 0 et 319 815 679.
Asymétrie d'UC	`cpu_skew`	Ratio de l'utilisation maximale de l'UC pour une tranche afin d'obtenir l'utilisation moyenne de l'UC pour toutes les tranches. Cette métrique est définie au niveau du segment. Les valeurs valides sont comprises entre 0 et 99.
Asymétrie d'I/O	`io_skew`	Ratio du nombre maximal de blocs lus (I/O) pour une tranche quelconque afin d'obtenir le nombre moyen de blocs lus pour toutes les tranches. Cette métrique est définie au niveau du segment. Les valeurs valides sont comprises entre 0 et 99.

Métrique	Name (Nom)	Description
Lignes jointes	`join_row_count`	Nombre de lignes traitées dans une étape de jonction. Les valeurs valides sont comprises entre 0 et 999 999 999 999 999.
Nombre de lignes jointes de boucle imbriquée	`nested_loop_join_row_count`	Nombre de lignes dans une jonction de boucles imbriquées. Les valeurs valides sont comprises entre 0 et 999 999 999 999 999.
Nombre de lignes renvoyées	`return_row_count`	Nombre de lignes retournées par la requête. Les valeurs valides sont comprises entre 0 et 999 999 999 999 999.
Durée d'exécution de segment	`segment_execution_time`	Délai écoulé pour l'exécution d'un seul segment (en secondes). Pour éviter ou réduire les risques d'erreurs d'échantillonnage, incluez `segment_execution_time > 10` à vos règles. Les valeurs valides sont comprises entre 0 et 86 388.
Nombre de lignes d'analyse Spectrum	`spectrum_scan_row_count`	Nombre de lignes de données dans Amazon S3 analysées par une requête Amazon Redshift Spectrum. Les valeurs valides sont comprises entre 0 et 999 999 999 999 999.

Métrique	Name (Nom)	Description
Taille d'analyse Spectrum	`spectrum_scan_size_mb`	Taille des données dans Amazon S3, en Mo, analysées par une requête Amazon Redshift Spectrum. Les valeurs valides sont comprises entre 0 et 999 999 999 999 999.
Priorité de requête	`query_priority`	Priorité de la requête. Les valeurs valides sont `HIGHEST`, `HIGH`, `NORMAL`, `LOW` et `LOWEST`. Si on compare `query_priority` à l'aide des signes supérieur à (>) et inférieur à (<), `HIGHEST` est supérieur à `HIGH`, `HIGH` est supérieur à `NORMAL`, etc.

> **Note**
>
> - L'action de saut n'est pas prise en charge avec le prédicat `query_queue_time`. Autrement dit, les règles définies pour le saut lorsqu'un prédicat `query_queue_time` est atteint sont ignorées.
> - La courte durée d'exécution de segment peut entraîner des erreurs d'échantillonnage avec certaines métriques, notamment `io_skew` et `query_cpu_usage_percent`. Pour éviter ou réduire les risques d'erreurs d'échantillonnage, incluez une durée d'exécution de segment à vos règles. `segment_execution_time > 10` vous donne un bon point de départ.

La vue SVL_QUERY_METRICS affiche les métriques des requêtes terminées. La vue SVL_QUERY_METRICS_SUMMARY affiche les valeurs maximales des métriques des requêtes terminées. Utilisez les valeurs de ces vues afin de vous aider à déterminer les seuils permettant de définir les règles de surveillance des requêtes.

Métriques de surveillance des requêtes pour Amazon Redshift sans serveur

La table suivante décrit les métriques utilisées dans les règles de surveillance de requête pour Amazon Redshift sans serveur.

Métrique	Name (Nom)	Description
Blocs lus	`max_query_blocks_read`	Nombre de blocs de données d'1 Mo lus par la requête. Les valeurs valides sont comprises entre 0 et 1 048 575.
Nombre de lignes d'analyse	`max_scan_row_count`	Nombre de lignes dans une étape d'analyse. Le nombre de lignes correspond au nombre total de lignes émises avant le filtrage des lignes marquées pour la suppression (lignes fantôme) et avant l'application des filtres de requête définis par l'utilisateur. Les valeurs valides sont comprises entre 0 et 999 999 999 999 999.
Durée d'exécution de la requête	max_query_execution_time	Délai écoulé pour l'exécution d'une requête (en secondes). Le délai d'exécution n'inclut pas le temps d'attente dans une file d'attente . Si une requête dépasse le délai d'exécution défini, Amazon Redshift sans serveur arrête la requête. Les valeurs valides sont comprises entre 0 et 86 399.
Temps de file d'attente de la requête	`max_query_queue_time`	Temps passé à attendre dans une file d'attente , en secondes. Les valeurs valides sont comprises entre 0 et 86 399.

Métrique	Name (Nom)	Description
Mémoire sur disque	`max_query_temp_blocks_to_disk`	Espace disque temporairement utilisé pour écrire des résultats intermédiaires, en blocs d'1 Mo. Les valeurs valides sont comprises entre 0 et 319 815 679.
Lignes jointes	`max_join_row_count`	Nombre de lignes traitées dans une étape de jonction. Les valeurs valides sont comprises entre 0 et 999 999 999 999 999.
Nombre de lignes jointes de boucle imbriquée	`max_nested_loop_join_row_count`	Nombre de lignes dans une jonction de boucles imbriquées. Les valeurs valides sont comprises entre 0 et 999 999 999 999 999.

> **Note**
>
> - L'action de saut n'est pas prise en charge avec le prédicat `max_query_queue_time`. Autrement dit, les règles définies pour le saut lorsqu'un prédicat `max_query_queue_time` est atteint sont ignorées.
> - La courte durée d'exécution de segment peut entraîner des erreurs d'échantillonnage avec certaines métriques, notamment `max_io_skew` et `max_query_cpu_usage_percent`.

Modèles de règles de surveillance de requête

Lorsque vous ajoutez une règle à l'aide de la console Amazon Redshift, vous pouvez choisir de créer une règle à partir d'un modèle prédéfini. Amazon Redshift crée une règle avec un ensemble de prédicats auxquels sont attribuées les valeurs par défaut. L'action par défaut est log. Vous pouvez modifier les prédicats et l'action en fonction de votre cas d'utilisation.

Le tableau suivant répertorie les modèles disponibles.

Nom du modèle	Prédicats	Description
Jointure de boucle imbriquée	`nested_loop_join_row_count > 100`	Une jonction de boucles imbriquées peut correspondre à un prédicat de jonction incomplet, qui se traduit généralement par un très grand nombre de retours (un produit cartésien). Utilisez un nombre de lignes peu élevé afin de détecter très tôt une potentielle requête d'échappement.
La requête renvoie un grand nombre de lignes	`return_row_count > 1000000`	Si une file d'attente est dédiée aux requêtes simples de courte durée, vous pouvez inclure une règle qui détecte les requêtes renvoyant un nombre élevé de lignes. Par défaut, le modèle utilise 1 million de lignes. Le nombre de lignes pouvant être désigné comme élevé dépend du système : ce sera un million de lignes sur certains systèmes, un milliard voire plus sur d'autres.
Jointure avec un grand nombre de lignes	`join_row_count > 1000000000`	Une étape de jonction qui implique un nombre de lignes anormalement élevé peut signifier qu'il convient d'utiliser des filtres plus restrictifs. Par défaut, le modèle utilise 1 milliard de lignes. Pour une file d'attente ad hoc (ponctuelle) destinée à des requêtes rapides et simples, vous pouvez utiliser un nombre inférieur.
Utilisation du disque élevée lors de l'écriture des résultats intermédiaires	`query_temp_blocks_to_disk > 100000`	Lorsque les requêtes en cours d'exécution utilisent davantage que la mémoire RAM système disponible, le moteur d'exécution des requêtes écrit les résultats intermédiaires sur le disque (mémoire déversée). En général, cette condition dérive d'une requête non autorisée, qui est habituellement la requête qui consomme le plus d'espace disque. Le seuil

Nom du modèle	Prédicats	Description
		acceptable d'utilisation du disque varie selon le type et le nombre des nœuds de cluster. Par défaut, le modèle utilise 100 000 blocs ou 100 Go. Si le cluster est petit, vous pouvez utiliser un nombre inférieur.
Requête longue avec un I/O biais élevé	`segment_execution_time > 120` et `io_skew > 1.30`	I/O skew occurs when one node slice has a much higher I/O rate than the other slices. As a starting point, a skew of 1.30 (1.3 times average) is considered high. High I/Ol'asymétrie n'est pas toujours un problème, mais lorsqu'elle est associée à une longue durée de requête, elle peut indiquer un problème lié au style de distribution ou à la clé de tri.

Tables et vues système pour les règles de surveillance de requête

Lorsque l'ensemble des prédicats d'une règle sont respectés, WLM écrit une ligne dans la table système STL_WLM_RULE_ACTION. Cette ligne contient les informations relatives à la requête qui a déclenché la règle et l'action qui en résulte.

En outre, Amazon Redshift enregistre les métriques des requêtes dans les tables système et les vues suivantes.

- Le tableau STV_QUERY_METRICS affiche les métriques des requêtes en cours d'exécution.
- Le tableau STL_QUERY_METRICS enregistre les métriques des requêtes terminées.
- La vue SVL_QUERY_METRICS affiche les métriques des requêtes terminées.
- La vue SVL_QUERY_METRICS_SUMMARY affiche les valeurs maximales des métriques des requêtes terminées.

Tables et vues système WLM

WLM configure les files d'attente de requêtes en fonction des classes de service WLM, qui sont définies en interne. Amazon Redshift crée plusieurs files d'attente internes en fonction de ces classes

de service, ainsi que des files d'attente définies dans la configuration de WLM. Les termes file d'attente et classe de service sont souvent utilisés indifféremment dans les tables système. La file d'attente du super-utilisateur utilise la classe de service 5. Les files d'attente définies par l'utilisateur utilisent la classe de service 6 et plus.

Vous pouvez afficher le statut de requêtes, des files d'attente et des classes de service à l'aide de tables système WLM spécifiques. Interrogez les tables système suivantes pour effectuer les opérations suivantes :

- Afficher les requêtes dont le suivi est effectué et les ressources qui sont allouées par le responsable de la charge de travail.
- Afficher à quelle file d'attente une requête a été affectée.
- Afficher le statut d'une requête qui est suivie actuellement par le responsable de la charge de travail.

Nom de la table	Description
STL_WLM_ERROR	Contient un journal des événements d'erreurs liés à WLM.
STL_WLM_QUERY	Répertorie les requêtes qui sont suivies par WLM.
STV_WLM_CLASSIFICATION_CONFIG	Affiche les règles de classification actuelles pour WLM.
STV_WLM_QUERY_QUEUE_STATE	Enregistre l'état actuel des files d'attente des requêtes.
STV_WLM_QUERY_STATE	Fournit un instantané de l'état actuel des requêtes qui sont suivies par WLM.
STV_WLM_QUERY_TASK_STATE	Contient l'état actuel des tâches des requêtes.
STV_WLM_SERVICE_CLASS_CONFIG	Enregistre les configurations de classe de service pour WLM.
STV_WLM_SERVICE_CLASS_STATE	Contient l'état actuel des classes de service.

Nom de la table	Description
STL_WLM_RULE_ACTION	Enregistre des détails relatifs aux actions résultant des règles de surveillance de requête WLM associées aux files d'attente définies par l'utilisateur.
STV_WLM_QMR_CONFIG	Enregistre la configuration des règles de surveillance de requête (QRM) WLM.

L'ID de tâche vous permet d'effectuer le suivi d'une requête dans les tables système. L'exemple suivant illustre comment obtenir l'ID de tâche de la requête d'utilisateur soumise le plus récemment :

```
select task from stl_wlm_query where exec_start_time =(select max(exec_start_time) from
 stl_wlm_query);

task
------
137
(1 row)
```

L'exemple suivant affiche les requêtes qui sont en cours ou en attente d'exécution dans différentes classes de service (files d'attente). Cette requête est utile pour suivre la charge de travail simultanée globale pour Amazon Redshift :

```
select * from stv_wlm_query_state order by query;


xid |task|query|service_| wlm_start_  |  state  |queue_ | exec_
    |    |     |class   | time        |         |time   | time
----+----+-----+--------+-------------+---------+-------+--------
2645| 84 | 98  | 3      | 2010-10-... |Returning|   0   | 3438369
2650| 85 | 100 | 3      | 2010-10-... |Waiting  |   0   | 1645879
2660| 87 | 101 | 2      | 2010-10-... |Executing|   0   | 916046
2661| 88 | 102 | 1      | 2010-10-... |Executing|   0   | 13291
(4 rows)
```

Classe de service WLM IDs

Le tableau suivant répertorie les classes de service IDs attribuées.

ID	Classe de service
1 – 4	Réservé au système.
5	Utilisé par la file d'attente de super-utilisateur.
6 – 13	Utilisé par les files d'attente de la gestion manuelle de la charge de travail définies dans la configuration WLM.
14	Utilisé par l'accélération des requêtes courtes.
15	Réservé pour les activités de maintenance exécutées par Amazon Redshift.
100 – 107	Utilisé par la file d'attente de la gestion automatique de la charge de travail lorsque auto_wlm a la valeur true.

Sécurité des bases de données

Vous pouvez gérer de la sécurité de la base de données en contrôlant les utilisateurs ayant accès aux objets qu'elle contient. Les utilisateurs peuvent se voir attribuer des rôles ou des groupes, et les autorisations que vous accordez aux utilisateurs, aux rôles ou aux groupes déterminent les objets de base de données auxquels ils peuvent accéder.

Rubriques

- Présentation de la sécurité d'Amazon Redshift
- Autorisations par défaut des utilisateurs de base de données
- Super-utilisateurs
- Users
- Groups
- Schémas
- Contrôle d'accès basé sur les rôles (RBAC)
- Sécurité au niveau des lignes
- Sécurité des métadonnées
- Masquage dynamique des données
- Autorisations étendues

L'accès aux objets de base de données dépend des autorisations que vous accordez aux utilisateurs ou aux rôles. Les consignes suivantes résument le fonctionnement de la sécurité de la base de données :

- Par défaut, les autorisations sont accordées uniquement au propriétaire de l'objet.
- Les utilisateurs de la base de données Amazon Redshift sont appelés des utilisateurs qui peuvent se connecter à une base de données. Un utilisateur bénéficie des autorisations de deux façons : de manière explicite, en affectant ces autorisations directement au compte, ou implicitement, en étant membre d'un groupe qui bénéficie d'autorisations.
- Les groupes sont des ensembles d'utilisateurs qui peuvent bénéficier collectivement d'autorisations à des fins de maintenance de sécurité rationalisée.
- Les schémas sont des ensembles de tables de base de données et d'autres objets de base de données. Les schémas sont similaires aux répertoires de système de fichiers, sauf qu'ils ne

peuvent pas s'imbriquer. Les utilisateurs peuvent bénéficier d'un accès à un seul schéma ou à plusieurs schémas.

En outre, Amazon Redshift utilise les fonctionnalités suivantes pour vous permettre de contrôler plus finement quels utilisateurs ont accès à quels objets de la base de données :

- Le contrôle d'accès basé sur les rôles (RBAC) vous permet d'attribuer des autorisations à des rôles que vous pouvez ensuite appliquer à des utilisateurs, ce qui vous permet de contrôler les autorisations pour de grands groupes d'utilisateurs. Contrairement aux groupes, les rôles peuvent hériter des autorisations d'autres rôles.

  La sécurité au niveau des lignes (RLS) vous permet de définir des stratégies qui restreignent l'accès aux lignes de votre choix, puis d'appliquer ces stratégies à des utilisateurs ou à des groupes.

  Le masquage dynamique des données (DDM) protège davantage vos données en les transformant au moment de l'exécution de la requête, de sorte que vous puissiez permettre aux utilisateurs d'accéder aux données sans exposer de détails sensibles.

Pour obtenir des exemples d'implémentation de la sécurité, consultez Exemple de contrôle d'accès utilisateur et de groupe.

Pour plus d'informations sur la protection de vos données, consultez Sécurité d'Amazon Redshift du Guide de la gestion du cluster Amazon Redshift.

Présentation de la sécurité d'Amazon Redshift

La sécurité de base de données Amazon Redshift est distincte des autres types de sécurité Amazon Redshift. Outre la sécurité de la base de données, décrite dans cette section, Amazon Redshift fournit les fonctions de gestion de la sécurité suivantes :

- Informations de connexion : l'accès à votre console de gestion Amazon AWS Redshift est contrôlé par les autorisations de AWS votre compte. Pour plus d'informations, consultez Informations d'identification de connexion.
- Gestion des accès : pour contrôler l'accès à des ressources Amazon Redshift spécifiques, vous devez définir des comptes AWS Identity and Access Management (IAM). Pour plus d'informations, consultez Contrôle de l'accès aux ressources Amazon Redshift.

- Groupes de sécurité du cluster : pour accorder à d'autres utilisateurs l'accès entrant à un cluster Amazon Redshift, vous devez définir un groupe de sécurité du cluster et l'associer à un cluster. Pour de plus amples informations, consultez Groupes de sécurité du cluster Amazon Redshift.
- VPC : pour protéger l'accès à votre cluster à l'aide d'un environnement de réseau virtuel, vous pouvez lancer votre cluster dans un cloud privé virtuel (VPC) Amazon. Pour plus d'informations, consultez Gestion des clusters dans un cloud privé virtuel (VPC).
- Chiffrement du cluster : pour chiffrer les données de toutes vos tables créées par l'utilisateur, vous pouvez activer le chiffrement du cluster au lancement de celui-ci. Pour plus d'informations, consultez Clusters Amazon Redshift.
- Connexions SSL : pour chiffrer la connexion entre votre client SQL et votre cluster, vous pouvez utiliser le chiffrement SSL (Secure Sockets Layer). Pour plus d'informations, consultez Connexion à votre cluster à l'aide de SSL.
- Chiffrement des données de chargement : pour chiffrer les fichiers de données de chargement de votre table lorsque vous les chargez dans Amazon S3, vous pouvez utiliser le chiffrement côté serveur ou chiffrement côté client. Lorsque vous chargez des données chiffrées côté serveur, Amazon S3 gère le déchiffrement de manière transparente. Lorsque vous chargez des données chiffrées côté client, la commande COPY Amazon Redshift déchiffre les données à mesure qu'elle charge la table. Pour de plus amples informations, veuillez consulter Chargement de données chiffrées sur Amazon S3.
- Données en transit : pour protéger vos données en transit dans le AWS cloud, Amazon Redshift utilise le protocole SSL à accélération matérielle pour communiquer avec Amazon S3 ou Amazon DynamoDB pour les opérations de copie, de déchargement, de sauvegarde et de restauration.
- Contrôle d'accès de niveau colonne : pour bénéficier d'un contrôle d'accès de niveau colonne pour les données dans Amazon Redshift, utilisez des instructions d'autorisation et de révocation de niveau colonne sans avoir à implémenter un contrôle d'accès basé sur les vues ou utiliser un autre système.
- Contrôle de sécurité au niveau des lignes : pour contrôler la sécurité des données au niveau des lignes dans Amazon Redshift, créez et associez des politiques aux rôles ou aux utilisateurs qui limitent l'accès aux lignes définies dans la politique.

Autorisations par défaut des utilisateurs de base de données

Lorsque vous créez un objet de base de données, vous en êtes le propriétaire. Par défaut, seul un super-utilisateur ou le propriétaire d'un objet peut exécuter une requête, apporter des modifications ou accorder des autorisations sur cet objet. Pour qu'un utilisateur puisse utiliser un objet, vous devez

accorder les autorisations nécessaires à cet utilisateur ou au groupe auquel il appartient. Les super-utilisateurs de base de données disposent des mêmes autorisations que les propriétaires de base de données.

Amazon Redshift prend en charge les autorisations suivantes : SELECT, INSERT, UPDATE, DELETE, REFERENCES, CREATE, TEMPORARY et USAGE. Des autorisations distinctes sont associées à différents types d'objets. Pour en savoir plus sur les autorisations d'objet de base de données prises en charge par Amazon Redshift, consultez la commande GRANT.

Seul le propriétaire a le droit de modifier ou de détruire un objet.

Par défaut, tous les utilisateurs disposent des autorisations CREATE et USAGE sur le schéma PUBLIC d'une base de données. Pour interdire aux utilisateurs de créer des objets dans le schéma PUBLIC d'une base de données, utilisez la commande REVOKE pour supprimer cette autorisation.

Pour annuler une autorisation qui a déjà été accordée, utilisez la commande REVOKE. Les autorisations du propriétaire de l'objet, telles que les autorisations DROP, GRANT et REVOKE, sont implicites et ne peuvent pas être accordées ou révoquées. Les propriétaires d'objets peuvent révoquer leurs propres autorisations ordinaires, par exemple, pour passer une table en lecture seule pour eux-mêmes et les autres. Les super-utilisateurs conservent toutes les autorisations, indépendamment des commandes GRANT et REVOKE.

Super-utilisateurs

Les super-utilisateurs de base de données disposent d'autorisations identiques à celles des propriétaires de base de données pour toutes les bases de données.

L'administrateur, qui est l'utilisateur que vous avez créé lorsque vous avez lancé le cluster, est un super-utilisateur.

Vous devez être un super-utilisateur pour créer un super-utilisateur.

Les tables et vues système Amazon Redshift sont visibles des super-utilisateurs ou de tous les utilisateurs. Seuls les super-utilisateurs peuvent interroger les tables et les vues système qui sont désignées comme « visibles des super-utilisateurs ». Pour plus d'informations, veuillez consulter Vues de surveillance SYS.

Les super-utilisateurs peuvent afficher toutes les tables de catalogue. Pour plus d'informations, veuillez consulter Tables catalogue système.

Un super-utilisateur de base de données contourne toutes les vérifications d'autorisations. Les super-utilisateurs conservent toutes les autorisations, indépendamment des commandes GRANT et REVOKE. Soyez prudent lorsque vous utilisez un rôle de super-utilisateur. Nous vous recommandons d'utiliser un rôle autre que celui de super-utilisateur lorsque cela est possible. Vous pouvez créer un rôle d'administrateur avec des autorisations plus restrictives. Pour plus d'informations sur la création de rôles, consultez Contrôle d'accès basé sur les rôles (RBAC).

Pour créer un nouveau super-utilisateur de base de données, connectez-vous à la base de données en tant que super-utilisateur et exécutez une commande CREATE USER ou ALTER USER avec l'autorisation CREATEUSER.

```
CREATE USER adminuser CREATEUSER PASSWORD '1234Admin';
ALTER USER adminuser CREATEUSER;
```

Pour créer, modifier ou supprimer un super-utilisateur, vous pouvez utiliser les commandes de gestion classiques des utilisateurs. Pour de plus amples informations, veuillez consulter Création, modification et suppression d'utilisateurs.

Users

Vous pouvez créer et gérer des utilisateurs de base de données à l'aide des commandes SQL Amazon Redshift CREATE USER et ALTER USER. Ou bien, vous pouvez configurer votre client SQL avec des pilotes personnalisés Amazon Redshift JDBC ou ODBC. Ces derniers gèrent la création d'utilisateurs de base de données et de mots de passe temporaires dans le cadre du processus de connexion à une base de données.

Les pilotes authentifient les utilisateurs de la base de données sur la base de l'authentification AWS Identity and Access Management (IAM). Si vous gérez déjà les identités des utilisateurs en dehors de AWS, vous pouvez utiliser un fournisseur d'identité (IdP) compatible SAML 2.0 pour gérer l'accès aux ressources Amazon Redshift. Vous utilisez un rôle IAM pour configurer votre IdP AWS et pour permettre à vos utilisateurs fédérés de générer des informations d'identification temporaires et de se connecter aux bases de données Amazon Redshift. Pour plus d'informations, consultez Utilisation de l'authentification IAM pour générer des informations d'identification de l'utilisateur de base de données.

Les utilisateurs Amazon Redshift peuvent uniquement être créés et supprimés par un super-utilisateur de base de données. Les utilisateurs sont authentifiés lorsqu'ils se connectent à Amazon Redshift. Ils peuvent posséder des bases de données et des objets de base de données

(par exemple, des tables). Ils peuvent également accorder à des utilisateurs, des groupes et des schémas des autorisations sur ces objets afin de contrôler l'accès aux objets. Les utilisateurs disposant de droits CREATE DATABASE peuvent créer des bases de données et accorder des autorisations concernant ces bases de données. Les super-utilisateurs disposent d'autorisations de propriétaires de bases de données pour toutes les bases de données.

Création, modification et suppression d'utilisateurs

Les utilisateurs de base de données appartiennent à un cluster d'entrepôt des données ; ils n'appartiennent pas à une base de données spécifique.

- Pour créer un utilisateur, utilisez la commande CREATE USER.
- Pour créer un super-utilisateur, utilisez la commande CREATE USER avec l'option CREATEUSER.
- Pour supprimer un utilisateur existant, utilisez la commande DROP USER.
- Pour modifier un utilisateur, par exemple modifier un mot de passe, utilisez la commande ALTER USER.
- Pour afficher une liste d'utilisateurs, interrogez la table de catalogue PG_USER.

```
select * from pg_user;

  username   | usesysid | usecreatedb | usesuper | usecatupd |  passwd  | valuntil |
 useconfig
------------+----------+-------------+----------+-----------+----------+----------
+-----------
 rdsdb      |        1 | t           | t        | t         | ******** |          |
 masteruser |      100 | t           | t        | f         | ******** |          |
 dwuser     |      101 | f           | f        | f         | ******** |          |
 simpleuser |      102 | f           | f        | f         | ******** |          |
 poweruser  |      103 | f           | t        | f         | ******** |          |
 dbuser     |      104 | t           | f        | f         | ******** |          |
(6 rows)
```

Groups

Les groupes sont des ensembles d'utilisateurs qui bénéficient tous des autorisations qui sont associées au groupe. Vous pouvez utiliser des groupes pour attribuer des autorisations. Par exemple, vous pouvez créer différents groupes pour les ventes, l'administration et le support et accorder aux utilisateurs de chaque groupe l'accès correspondant aux données qu'ils ont besoin d'utiliser.

Vous pouvez accorder ou révoquer des autorisations au niveau du groupe, et ces modifications s'appliqueront à tous les membres du groupe, à l'exception des super-utilisateurs.

Pour afficher tous les groupes d'utilisateurs, interrogez la table catalogue système PG_GROUP :

```
select * from pg_group;
```

Par exemple, pour répertorier tous les utilisateurs d'une base de données par groupe, exécutez la requête SQL suivante.

```
SELECT u.usesysid
,g.groname
,u.usename
FROM pg_user u
LEFT JOIN pg_group g ON u.usesysid = ANY (g.grolist)
```

Création, modification et suppression de groupes

Seul un super-utilisateur peut créer, modifier ou supprimer des groupes.

Vous pouvez effectuer les opérations suivantes :

- Pour créer un groupe, utilisez la commande CREATE GROUP.
- Pour ajouter ou supprimer des utilisateurs d'un groupe existant, utilisez la commande ALTER GROUP.
- Pour supprimer un groupe, utilisez la commande DROP GROUP. Cette commande supprime uniquement le groupe, pas les utilisateurs qui en sont membres.

Exemple de contrôle d'accès utilisateur et de groupe

Cet exemple crée des groupes d'utilisateurs et des utilisateurs, puis leur accorde différentes autorisations concernant une base de données Amazon Redshift qui se connecte à un client d'application web. Cet exemple suppose qu'il existe trois groupes d'utilisateurs : les utilisateurs habituels d'une application web, les utilisateurs avancés d'une application web et les développeurs web.

Pour plus d'informations sur la façon de supprimer un utilisateur d'un groupe, consultezALTER GROUP.

1. Créez les groupes dans lesquels les utilisateurs seront affectés. L'ensemble de commandes suivant crée trois groupes d'utilisateurs différents :

```
create group webappusers;

create group webpowerusers;

create group webdevusers;
```

2. Créez plusieurs utilisateurs de base de données avec différentes autorisations et ajoutez-les aux groupes.

   a. Créez deux utilisateurs et ajoutez-les au groupe WEBAPPUSERS :

```
create user webappuser1 password 'webAppuser1pass'
in group webappusers;

create user webappuser2 password 'webAppuser2pass'
in group webappusers;
```

   b. Créez un utilisateur développeur web et ajoutez-le au groupe WEBDEVUSERS :

```
create user webdevuser1 password 'webDevuser2pass'
in group webdevusers;
```

   c. Créez un super-utilisateur. Cet utilisateur disposera de droits d'administration lui permettant de créer d'autres utilisateurs :

```
create user webappadmin  password 'webAppadminpass1'
createuser;
```

3. Créez un schéma à associer aux tables de base de données utilisées par l'application web et accordez aux différents groupes d'utilisateurs l'accès à ce schéma :

   a. Créez le schéma WEBAPP :

```
create schema webapp;
```

   b. Accordez les autorisations USAGE au groupe WEBAPPUSERS :

```
grant usage on schema webapp to group webappusers;
```

   c. Accordez les autorisations USAGE au groupe WEBPOWERUSERS :

```
grant usage on schema webapp to group webpowerusers;
```

d. Accordez les autorisations ALL au groupe WEBDEVUSERS :

```
grant all on schema webapp to group webdevusers;
```

Les utilisateurs et les groupes de base sont maintenant configurés. Vous pouvez à présent apporter des modifications aux utilisateurs et aux groupes.

4. Par exemple, la commande suivante modifie le paramètre search_path pour le. WEBAPPUSER1

```
alter user webappuser1 set search_path to webapp, public;
```

Le paramètre SEARCH_PATH spécifie l'ordre de recherche de schéma des objets de base de données, tels que les tables et les fonctions, lorsque l'objet est référencé par un nom simple sans schéma spécifié.

5. Vous pouvez également ajouter des utilisateurs à un groupe après l'avoir créé, par exemple en les ajoutant WEBAPPUSER2 au groupe WEBPOWERUSERS :

```
alter group webpowerusers add user webappuser2;
```

Schémas

Une base de données contient un ou plusieurs schémas désignés. Chaque schéma dans une base de données contient des tables et d'autres types d'objets nommés. Par défaut, une base de données se compose d'un schéma unique, qui est appelé PUBLIC. Vous pouvez utiliser des schémas pour regrouper des objets de base de données sous un nom commun. Les schémas sont similaires aux répertoires de système de fichiers, sauf qu'ils ne peuvent pas s'imbriquer.

Des noms d'objet de base de données identiques peuvent être utilisés dans des schémas distincts de la même base de données sans que cela entraîne de conflits. Par exemple, MY_SCHEMA et YOUR_SCHEMA peuvent contenir une table nommée MYTABLE. Les utilisateurs disposant des autorisations nécessaires peuvent accéder aux objets sur plusieurs schémas dans une base de données.

Par défaut, un objet est créé dans le premier schéma du chemin d'accès de la base de données. Pour plus d'informations, consultez Chemin de recherche plus loin dans cette section.

Les schémas permettent de résoudre les problèmes d'organisation et de simultanéité dans un environnement multi-utilisateur comme suit :

- Pour permettre à de nombreux développeurs d'utiliser la même base de données sans interférer dans leurs travaux respectifs.
- Pour organiser les objets de base de données en groupes logiques afin de les rendre plus faciles à gérer.
- Pour fournir à des applications la possibilité de placer leurs objets dans des schémas distincts, afin que leurs noms ne créent pas de conflits avec les noms des objets utilisés par d'autres applications.

Chemin de recherche

Le chemin de recherche est défini dans le paramètre search_path avec une liste séparée par des virgules de noms de schémas. Le chemin d'accès spécifie l'ordre dans lequel les schémas sont recherchés lorsqu'un objet, comme une table ou une fonction, est référencé par un nom simple qui n'inclut pas de qualificateur de schéma.

Si un objet est créé sans spécifier de schéma cible, l'objet est ajouté au premier schéma qui est répertorié dans le chemin de recherche. Lorsqu'il existe des objets avec des noms identiques dans des schémas distincts, un nom d'objet qui ne spécifie pas de schéma fera référence au premier schéma du chemin d'accès qui contient un objet portant ce nom.

Pour modifier le schéma par défaut de la session en cours, utilisez la commande SET.

Pour plus d'informations, consultez la description de search_path dans la Référence de configuration.

Création, modification et suppression de schémas

N'importe quel utilisateur peut créer des schémas et modifier ou supprimer ceux qu'il possède.

Vous pouvez effectuer les opérations suivantes :

- Pour créer un schéma, utilisez la commande CREATE SCHEMA.
- Pour modifier le propriétaire d'un schéma, utilisez la commande ALTER SCHEMA.
- Pour supprimer un schéma et ses objets, utilisez la commande DROP SCHEMA.
- Pour créer une table dans un schéma, créez-la au format schema_name.table_name.

Pour afficher une liste de tous les schémas, interrogez la table catalogue système PG_NAMESPACE :

```
select * from pg_namespace;
```

Pour afficher la liste des tables qui font partie d'un schéma, interrogez la table catalogue système PG_TABLE_DEF. Par exemple, la requête suivante renvoie une liste de tables dans le schéma PG_CATALOG.

```
select distinct(tablename) from pg_table_def
where schemaname = 'pg_catalog';
```

Autorisations basées un schéma

Les autorisations basées sur un schéma sont déterminées par le propriétaire du schéma :

- Par défaut, tous les utilisateurs disposent des autorisations CREATE et USAGE sur le schéma PUBLIC d'une base de données. Pour interdire aux utilisateurs de créer des objets dans le schéma PUBLIC d'une base de données, utilisez la commande REVOKE pour supprimer cette autorisation.
- Les utilisateurs ne peuvent accéder à aucun objet des schémas qui ne leur appartiennent pas, sauf s'ils bénéficient de l'autorisation USAGE accordée par le propriétaire de l'objet.
- Si les utilisateurs bénéficient de l'autorisation CREATE pour un schéma créé par un autre utilisateur, ils peuvent créer des objets dans ce schéma.

Contrôle d'accès basé sur les rôles (RBAC)

En utilisant le contrôle d'accès basé sur les rôles (RBAC) afin de gérer les autorisations de base de données dans Amazon Redshift, vous pouvez simplifier la gestion des autorisations de sécurité dans Amazon Redshift. Vous pouvez sécuriser l'accès aux données sensibles en contrôlant ce que les utilisateurs peuvent faire à un niveau étendu ou plus précis. Vous pouvez également contrôler l'accès des utilisateurs aux tâches normalement limitées aux super-utilisateurs. En attribuant différentes autorisations à différents rôles et en les attribuant à différents utilisateurs, vous pouvez bénéficier d'un contrôle plus précis de l'accès des utilisateurs.

Les utilisateurs dotés d'un rôle assigné peuvent uniquement effectuer les tâches spécifiées par le rôle assigné auquel ils sont autorisés. Par exemple, un utilisateur doté du rôle assigné ayant les autorisations CREATE TABLE et DROP TABLE est uniquement autorisé à effectuer ces tâches. Vous

pouvez contrôler l'accès des utilisateurs en accordant différents niveaux d'autorisations de sécurité à différents utilisateurs afin qu'ils puissent accéder aux données dont ils ont besoin pour leur travail.

Le RBAC applique le principe des moindres autorisations aux utilisateurs en fonction de leurs exigences de rôle, quels que soient les types d'objets concernés. L'octroi et la révocation des autorisations s'effectuent au niveau du rôle, sans qu'il soit nécessaire de mettre à jour les autorisations sur des objets de base de données individuels.

Avec le RBAC, vous pouvez créer des rôles dotés d'autorisations permettant d'exécuter des commandes qui nécessitaient auparavant des autorisations de super-utilisateur. Les utilisateurs peuvent exécuter ces commandes tant qu'un rôle incluant ces autorisations leur est autorisé. De même, vous pouvez également créer des rôles afin de limiter l'accès à certaines commandes et attribuer le rôle à des super-utilisateurs ou à des utilisateurs pour lesquels le rôle a été autorisé.

Pour en savoir plus sur le contrôle d'accès basé sur les rôles (RBAC) Amazon Redshift, regardez la vidéo suivante :Présentation du contrôle d'accès basé sur les rôles (RBAC) dans Amazon Redshift.

Hiérarchie des rôles

Les rôles sont des ensembles d'autorisations que vous pouvez attribuer à un utilisateur ou à un autre rôle. Vous pouvez attribuer des autorisations système ou de base de données à un rôle. Un utilisateur hérite des autorisations d'un rôle assigné.

Dans le RBAC, les utilisateurs peuvent disposer de rôles imbriqués. Vous pouvez attribuer des rôles à la fois à des utilisateurs et à des rôles. Lorsque vous accordez un rôle à un utilisateur, vous autorisez l'utilisateur à disposer de toutes les autorisations incluses dans ce rôle. Lorsque vous accordez un rôle r1 à un utilisateur, vous autorisez l'utilisateur à disposer des autorisations de r1. L'utilisateur dispose désormais des autorisations de r1 et de toutes les autorisations existantes qu'il possède déjà.

Lorsque vous accordez un rôle (r1) à un autre rôle (r2), vous autorisez r2 à disposer de toutes les autorisations de r1. En outre, lors de l'octroi de r2 à un autre rôle (r3), les autorisations de r3 équivalent à la combinaison des autorisations de r1 et de r2. La hiérarchie des rôles sous-entend que r2 hérite des autorisations de r1. Amazon Redshift propage les autorisations à chaque autorisation de rôle. L'octroi de r1 à r2, puis de r2 à r3 a pour effet d'autoriser r3 à disposer de toutes les autorisations des trois rôles. Ainsi, en accordant r3 à un utilisateur, l'utilisateur dispose de toutes les autorisations des trois rôles.

Amazon Redshift n'autorise pas la création d'un cycle d'autorisation des rôles. Un cycle d'autorisation des rôles se crée lorsqu'un rôle imbriqué est réaffecté à un rôle se trouvant plus avant dans la

hiérarchie des rôles, tel que le fait de réassigner r3 à r1. Pour plus d'informations sur la création de rôles et la gestion des affectations de rôles, consultez Gestion des rôles dans le RBAC.

Attribution des rôles

Les super-utilisateurs et les utilisateurs standard disposant des autorisations CREATE ROLE peuvent utiliser l'instruction CREATE ROLE afin de créer des rôles. Les super-utilisateurs et les administrateurs de rôle peuvent utiliser l'instruction GRANT ROLE pour accorder un rôle aux autres. Ils peuvent utiliser l'instruction REVOKE ROLE pour révoquer un rôle assigné à d'autres, et l'instruction DROP ROLE pour supprimer des rôles. Les administrateurs de rôles incluent les propriétaires de rôles et les utilisateurs auxquels le rôle a été accordé avec l'autorisation ADMIN OPTION.

Seuls les super-utilisateurs ou les administrateurs de rôle peuvent accorder et révoquer des rôles. Vous pouvez accorder un ou plusieurs rôles à un ou plusieurs rôles ou utilisateurs, ou révoquer un ou plusieurs rôles assignés à un ou plusieurs rôles ou utilisateurs. Utilisez l'option WITH ADMIN OPTION de l'instruction GRANT ROLE afin de fournir les options d'administration de tous les rôles accordés à tous les bénéficiaires.

Amazon Redshift prend en charge différentes combinaisons d'attributions de rôles, telles que l'octroi de plusieurs rôles ou l'assignation de plusieurs bénéficiaires. L'option WITH ADMIN OPTION s'applique uniquement aux utilisateurs, et non aux rôles. De même, utilisez l'option WITH ADMIN OPTION de l'instruction REVOKE ROLE pour supprimer le rôle et l'autorisation administrative du bénéficiaire. Lorsqu'elle est utilisée avec l'option ADMIN OPTION, seule l'autorisation administrative est révoquée pour le rôle.

L'exemple suivant révoque l'autorisation administrative du rôle `sample_role2` pour `user2`.

```
REVOKE ADMIN OPTION FOR sample_role2 FROM user2;
```

Pour plus d'informations sur la création de rôles et la gestion des affectations de rôles, consultez Gestion des rôles dans le RBAC.

Rôles définis par le système Amazon Redshift

Amazon Redshift fournit quelques rôles définis par le système qui sont définis au moyen d'autorisations spécifiques. Les rôles propres au système commencent par un préfixe `sys:`. Seuls les utilisateurs disposant d'un accès approprié peuvent modifier des rôles définis par le système ou

créer des rôles personnalisés définis par le système. Vous ne pouvez pas utiliser le préfixe `sys:` pour un rôle personnalisé défini par le système.

Le tableau suivant résume les rôles et leurs autorisations.

Nom du rôle	Description
`sys:monitor`	Ce rôle dispose des autorisations nécessaires pour accéder aux tables système ou du catalogue.
`sys:operator`	Ce rôle dispose des autorisations nécessaires pour accéder aux tables catalogue ou système, analyser, vider ou annuler des requêtes.
`sys:dba`	Ce rôle dispose des autorisations nécessaires pour créer des schémas, créer des tables, supprimer des schémas, supprimer des tables et tronquer des tables. Il dispose des autorisations nécessaires pour créer ou remplacer des procédures stockées, supprimer des procédures,

Nom du rôle	Description		
	créer ou remplacer des fonctions, créer ou remplacer des fonctions externes, créer des vues, et supprimer des vues. De plus, ce rôle hérite de toutes les autorisations du rôle sys:operator.		
`sys:superuser`	Ce rôle possède tous les autorisations système prise en charge définis dans Autorisations système pour le RBAC.		

Nom du rôle	Description		
`sys:secadmin`	• Ce rôle dispose des autorisations nécessaires pour créer des utilisateurs, modifier des utilisateurs, supprimer des utilisateurs, créer des rôles, supprimer des rôles et accorder des rôles. • Ce rôle est autorisé à activer (ON) ou à désactiver (OFF) le protocole RLS sur une relation et à gérer les politiques RLS et DDM (CREATE, DROP, ATTACH, DETACH et ALTER). Notez également que les autorisations EXPLAIN RLS, IGNORE RLS et EXPLAIN MASKING sont accordées par défaut à ce rôle. • Ce rôle peut avoir accès aux tables utilisateur uniquement		

Nom du rôle	Description		
	lorsque l'autorisation est explicitement accordée au rôle.		

Rôles et utilisateurs définis par le système pour le partage des données

Amazon Redshift crée des rôles et des utilisateurs à usage interne qui correspondent aux partages de données et aux consommateurs de partages de données. Chaque nom de rôle interne et nom d'utilisateur possède le préfixe `ds:` d'espace de noms réservé. Ils ont le format suivant :

Name (Nom)	Description		
`ds:`*`sharename`*	Rôle système correspondant à un partage de données.		
`ds:`*`sharename_consumer`*	Un utilisateur du système qui correspond à un consommateur de partage de données.		

Un rôle de partage de données est créé pour chaque partage de données. Il contient toutes les autorisations actuellement accordées au partage de données. Un utilisateur de partage de données est créé pour chaque consommateur d'un partage de données. Il est autorisé à utiliser un seul rôle de partage de données. Un consommateur ajouté à plusieurs partages de données aura un utilisateur de partage de données créé pour chaque partage de données.

Ces utilisateurs et rôles sont nécessaires au bon fonctionnement du partage de données. Ils ne peuvent pas être modifiés ou supprimés et ils ne sont pas accessibles ou utilisés pour aucune tâche exécutée par les clients. Vous pouvez les ignorer en toute sécurité. Pour plus d'informations sur le partage de données, consultez Partage de données entre clusters dans Amazon Redshift.

> **Note**
>
> Vous ne pouvez pas utiliser le `ds:` préfixe pour créer des rôles ou des utilisateurs définis par l'utilisateur.

Autorisations système pour le RBAC

Vous trouverez ci-dessous une liste d'autorisations système que vous pouvez accorder à un rôle ou révoquer pour celui-ci.

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
CREATE ROLE	• Super-utilisateur. • Utilisateurs disposant de l'autorisation CREATE ROLE.		
DROP ROLE	• Super-utilisateur. • Propriétaire du rôle qui est soit l'utilisateur qui a créé le rôle, soit un utilisateur qui s'est vu accorder le rôle avec l'autorisation WITH ADMIN OPTION.		
CREATE USER	• Super-utilisateur. • Utilisateurs disposant de		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
	l’autorisation CREATE USER. Ces utilisateurs ne peuvent pas créer de super-uti lisateurs.		
DROP USER	• Super-utilisateur. • Utilisateurs disposant de l’autorisation DROP USER.		
ALTER USER	• Super-utilisateur. • Utilisateurs disposant de l’autorisation ALTER USER. Ces utilisateurs ne peuvent pas modifier les utilisate urs en super-utilisateurs ou modifier les super-utilisateurs en utilisateurs. • Utilisateur actuel qui souhaite modifier son propre mot de passe.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
CREATE SCHEMA	• Super-utilisateur. • Utilisateurs disposant de l’autorisation CREATE SCHEMA.		
DROP SCHEMA	• Super-utilisateur. • Utilisateurs disposant de l’autorisation DROP SCHEMA. • Propriétaire du schéma.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
ALTER DEFAULT PRIVILEGES	• Super-utilisateur. • Utilisateurs disposant de l’autorisation ALTER DEFAULT PRIVILEGES. • Utilisateurs modifiant leurs propres autorisations d’accès par défaut. • Utilisateurs définissant des autorisations pour les schémas pour lesquels ils disposent d’autorisations d’accès.		
ACCESS CATALOG	• Super-utilisateur. • Utilisateurs disposant de l'autorisation ACCESS CATALOG.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
TABLE DU SYSTÈME D'ACCÈS	• Super-utilisateur. • Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE.		
CREATE TABLE	• Super-utilisateur. • Utilisateurs disposant de l’autorisation CREATE TABLE. • Utilisateurs disposant de l’autorisation CREATE sur les schémas.		
DROP TABLE	• Super-utilisateur. • Utilisateurs disposant de l’autorisation DROP TABLE. • Propriétaire de table disposant de l’autorisation USAGE sur le schéma.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
ALTER TABLE	• Super-utilisateur. • Utilisateurs disposant de l’autorisation ALTER TABLE. • Propriétaire de table disposant de l’autorisation USAGE sur le schéma.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		

CREATE OR REPLACE FUNCTION	• Pour CREATE FUNCTION :
	  • Super-utilisateur.
	  • Utilisateurs disposant de l’autorisation CREATE OR REPLACE FUNCTION.
	  • Utilisateurs disposant de l’autorisation USAGE sur le langage.
	• Pour REPLACE FUNCTION :
	  • Super-utilisateur.
	  • Utilisateurs disposant de l’autorisation CREATE OR REPLACE FUNCTION.
	  • Propriétaire de la fonction.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
CREATE OR REPLACE EXTERNAL FUNCTION	• Super-utilisateur. • Utilisateurs possédant l’autorisation CREATE OR REPLACE EXTERNAL FUNCTION.		
DROP FUNCTION	• Super-utilisateur. • Utilisateurs dotés de l’autorisation DROP FUNCTION. • Propriétaire de la fonction.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		

CREATE OR REPLACE PROCEDURE	• Pour CREATE PROCEDURE :
	  • Super-utilisateur.
	  • Utilisateurs disposant de l’autorisation CREATE OR REPLACE PROCEDURE.
	  • Utilisateurs disposant de l’autorisation USAGE sur le langage.
	• Pour REPLACE PROCEDURE :
	  • Super-utilisateur.
	  • Utilisateurs disposant de l’autorisation CREATE OR REPLACE PROCEDURE.
	  • Propriétaire de la procédure.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
DROP PROCEDURE	• Super-utilisateur. • Utilisateurs dotés de l’autorisation DROP PROCEDURE. • Propriétaire de la procédure.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		

CREATE OR REPLACE VIEW	• Pour CREATE VIEW :
	  • Super-utilisateur.
	  • Utilisateurs disposant de l’autorisation CREATE OR REPLACE VIEW.
	  • Utilisateurs disposant de l’autorisation CREATE sur les schémas.
	• Pour REPLACE VIEW :
	  • Super-utilisateur.
	  • Utilisateurs disposant de l’autorisation CREATE OR REPLACE VIEW.
	  • Propriétaire de la vue.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
DROP VIEW	• Super-utilisateur. • Utilisateurs disposant de l’autorisation DROP VIEW. • Propriétaire de la vue.		
CREATE MODEL	• Super-utilisateur. • Utilisateurs disposant de l’autorisation système CREATE MODEL, qui doivent pouvoir lire la relation de CREATE MODEL. • Utilisateurs disposant de l’autorisation CREATE MODEL.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
DROP MODEL	• Super-utilisateur. • Utilisateurs disposant de l’autorisation DROP MODEL. • Propriétaire du modèle. • Propriétaire du schéma.		
CREATE DATASHARE	• Super-utilisateur. • Utilisateurs disposant de l’autorisation CREATE DATASHARE. • Propriétaire de la base de données.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
ALTER DATASHARE	• Super-utilisateur. • Utilisateur disposant de l’autorisation ALTER DATASHARE. • Utilisateurs disposant de l’autorisation ALTER ou ALL sur l’unité de partage des données. • Pour ajouter des objets spécifiques à une unité de partage des données, ces utilisateurs doivent avoir l’autorisation sur les objets. Les utilisateurs doivent être les propriétaires des objets ou disposer des autorisations SELECT, USAGE ou ALL sur les objets.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
DROP DATASHARE	• Super-utilisateur. • Utilisateurs disposant de l’autorisation DROP DATASHARE. • Propriétaire de la base de données.		
CREATE LIBRARY	• Super-utilisateur. • Utilisateurs disposant de l’autorisation CREATE LIBRARY ou de l’autorisation du langage spécifié.		
DROP LIBRARY	• Super-utilisateur. • Utilisateurs dotés de l’autorisation DROP LIBRARY. • Propriétaire de la bibliothèque.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
ANALYSE	• Super-utilisateur. • Utilisateurs disposant de l’autorisation ANALYZE. • Propriétaire de la relation. • Propriétaire de la base de données avec qui la table est partagée.		
ANNULER	• Super-utilisateur annulant sa propre requête. • Super-utilisateur annulant la requête d’un utilisateur. • Utilisateurs disposant de l’autorisation CANCEL et annulant la requête d’un utilisateur. • Utilisateur annulant sa propre requête.		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
TRUNCATE TABLE	• Super-utilisateur. • Utilisateurs dotés de l’autorisation TRUNCATE TABLE. • Propriétaire de la table.		
VACUUM	• Super-utilisateur. • Utilisateurs disposant de l’autorisation VACUUM. • Propriétaire de la table. • Propriétaire de la base de données avec qui la table est partagée.		
IGNORE RLS	• Super-utilisateur. • Utilisateurs au sein du rôle `sys:secadmin`		
EXPLAIN RLS	• Super-utilisateur. • Utilisateurs au sein du rôle `sys:secadmin`		

Command	Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la commande		
EXPLAIN MASKING	• Super-utilisateur. • Utilisateurs au sein du rôle `sys:secadmin`		

Autorisations d’objet de base de données

Outre les autorisations système, Amazon Redshift comprend des autorisations d’objet de base de données qui définissent les options d’accès. Celles-ci incluent des options telles que la faculté de lire les données des tables et des vues, écrire des données, créer et supprimer des tables. Pour de plus amples informations, veuillez consulter GRANT.

En utilisant le RBAC, vous pouvez attribuer des autorisations d’objet de base de données à des rôles, de la même manière qu’avec les autorisations système. Vous pouvez ensuite attribuer des rôles à des utilisateurs, autoriser des utilisateurs disposant d’autorisations système et autoriser des utilisateurs disposant d’autorisations de base de données.

Instruction ALTER DEFAULT PRIVILEGES pour le RBAC

Utilisez l’instruction ALTER DEFAULT PRIVILEGES pour définir le groupe d’autorisations d’accès par défaut à appliquer sur des objets qui seront créés à l’avenir par l’utilisateur spécifié. Par défaut, les utilisateurs peuvent ne modifier que leurs propres autorisations d’accès par défaut. Avec le RBAC, vous pouvez définir les autorisations d’accès par défaut pour les rôles. Pour plus d’informations, consultez la commande ALTER DEFAULT PRIVILEGES.

Le RBAC vous permet d’attribuer des autorisations d’objet de base de données à des rôles, de la même manière qu’avec les autorisations système. Vous pouvez ensuite attribuer des rôles à des utilisateurs, autoriser des utilisateurs disposant d’autorisations système et/ou de base de données.

Considérations concernant l’utilisation des rôles dans le RBAC

Lorsque vous travaillez avec des rôles RBAC, tenez compte des points suivants :

- Amazon Redshift n'autorise pas les cycles d'autorisations des rôles. Vous ne pouvez pas accorder r1 à r2, puis accorder r2 à r1.
- Le RBAC fonctionne à la fois pour les objets Amazon Redshift natifs et les tables Amazon Redshift Spectrum.
- En tant qu'administrateur Amazon Redshift, vous pouvez activer le RBAC en mettant à niveau votre cluster vers le dernier correctif de maintenance pour commencer.
- Seuls les super-utilisateurs et les utilisateurs disposant de l'autorisation système CREATE ROLE peuvent créer des rôles.
- Seuls les super-utilisateurs et les administrateurs de rôle peuvent modifier ou supprimer des rôles.
- Un nom de rôle ne peut pas être identique à un nom d'utilisateur.
- Un nom de rôle ne peut pas contenir de caractères non valides, tels que « :/\n ».
- Un nom de rôle ne peut pas être un mot réservé, tel que PUBLIC.
- Le nom du rôle ne peut pas commencer par le préfixe réservé pour les rôles par défaut, à savoir `sys:`.
- Vous ne pouvez pas supprimer un rôle qui possède le paramètre RESTRICT lorsqu'il est accordé à un autre rôle. Le paramètre par défaut est RESTRICT. Amazon Redshift génère une erreur lorsque vous tentez de supprimer un rôle qui a hérité d'un autre rôle.
- Les utilisateurs qui ne disposent pas d'autorisations d'administrateur sur un rôle ne peuvent pas accorder ou révoquer un rôle.
- Le RBAC n'est pas entièrement pris en charge pour les tables et les vues système. Les autorisations RBAC pour les tables et les vues système ne sont pas conservées lors des mises à niveau, des rétrogradations ou des redimensionnements. Nous vous recommandons Rôles définis par le système Amazon Redshift de l'utiliser pour gérer la table système et afficher les autorisations. Pour plus d'informations sur les tables système, rendez-vous surInformations de référence sur les tables et les vues système.

Gestion des rôles dans le RBAC

Pour effectuer les actions suivantes, utilisez les commandes suivantes :

- Pour créer un rôle, utilisez la commande CREATE ROLE.
- Pour renommer un rôle ou modifier le propriétaire du rôle, utilisez la commande ALTER ROLE.
- Pour supprimer un rôle, utilisez la commande DROP ROLE.
- Pour attribuer un rôle à un utilisateur, utilisez la commande GRANT.

- Pour révoquer un rôle d'un utilisateur, utilisez la commande REVOKE.
- Pour accorder des autorisations système à un rôle, utilisez la commande GRANT.
- Pour révoquer des autorisations système pour un rôle, utilisez la commande REVOKE.

Pour afficher la liste des rôles de votre cluster ou de votre groupe de travail, consultez SVV_ROLES.

Tutoriel : Création de rôles et interrogation avec RBAC

Avec le RBAC, vous pouvez créer des rôles dotés d'autorisations permettant d'exécuter des commandes qui nécessitaient auparavant des autorisations de super-utilisateur. Les utilisateurs peuvent exécuter ces commandes tant qu'un rôle incluant ces autorisations leur est autorisé.

Dans ce didacticiel, vous allez utiliser le contrôle d'accès basé sur les rôles (RBAC) pour gérer les autorisations dans une base de données que vous créez. Vous vous connectez ensuite à la base de données et interrogez la base de données à partir de deux rôles différents pour tester les fonctionnalités du RBAC.

Les deux rôles que vous créez et utilisez pour interroger la base de données sont `sales_ro` et`sales_rw`. Vous créez le `sales_ro` rôle et interrogez les données en tant qu'utilisateur titulaire du `sales_ro` rôle. L'`sales_ro`utilisateur peut uniquement utiliser la commande SELECT mais ne peut pas utiliser la commande UPDATE. Vous créez ensuite le `sales_rw` rôle et interrogez les données en tant qu'utilisateur titulaire du `sales_rw` rôle. L'`sales_rw`utilisateur peut utiliser les commandes SELECT et UPDATE.

En outre, vous pouvez créer des rôles pour limiter l'accès à certaines commandes et attribuer le rôle à des superutilisateurs ou à des utilisateurs.

Tâches

- Prérequis
- Étape 1 : créer un utilisateur administrateur
- Étape 2 : Configuration des schémas
- Étape 3 : créer un utilisateur en lecture seule
- Étape 4 : interroger les données en tant qu'utilisateur en lecture seule
- Étape 5 : Création d'un utilisateur en lecture-écriture
- Étape 6 : Interrogez les données en tant qu'utilisateur avec le rôle en lecture seule hérité

- Étape 7 : Accorder des autorisations de mise à jour et d'insertion pour le rôle de lecture-écriture
- Étape 8 : Interrogez les données en tant qu'utilisateur en lecture-écriture
- Étape 9 : Analyser et vider les tables d'une base de données en tant qu'utilisateur administrateur
- Étape 10 : tronquer les tables en tant qu'utilisateur en lecture-écriture
- Fonctions du système pour RBAC (en option)
- Vues du système pour RBAC (en option)
- Utiliser la sécurité au niveau des lignes avec le RBAC (facultatif)

Prérequis

- Créez un cluster Amazon Redshift ou un groupe de travail sans serveur chargé avec la base de données d'exemples TICKIT. Pour créer un groupe de travail sans serveur, voir Commencer avec les entrepôts de données Redshift Serverless. Pour créer un cluster, consultez Créer un exemple de cluster Amazon Redshift. Pour plus d'informations sur l'exemple de base de données TICKIT, consultez Exemple de base de données.
- Accédez à un utilisateur doté d'autorisations de superutilisateur ou d'administrateur de rôles. Seuls les superutilisateurs ou les administrateurs de rôles peuvent octroyer ou révoquer des rôles. Pour plus d'informations sur les autorisations requises pour le RBAC, consultezAutorisations système pour le RBAC.
- Prenez connaissance des Considérations concernant l'utilisation des rôles dans le RBAC.

Étape 1 : créer un utilisateur administrateur

Pour configurer ce didacticiel, vous devez créer un rôle d'administrateur de base de données et l'associer à un utilisateur administrateur de base de données au cours de cette étape. Vous devez créer l'administrateur de base de données en tant que superutilisateur ou administrateur de rôles.

Exécutez toutes les requêtes dans l'éditeur de requêtes Amazon Redshift v2.

1. Pour créer le rôle d'administrateur db_admin, utilisez l'exemple suivant.

```
CREATE ROLE db_admin;
```

2. Pour créer un utilisateur de base de données nommé dbadmin, utilisez l'exemple suivant.

```
CREATE USER dbadmin PASSWORD 'Test12345';
```

3. Pour attribuer le rôle défini par le système nommé sys:dba au rôle db_admin, utilisez l'exemple suivant. Lorsque le rôle sys:dba est attribué, l'utilisateur dbadmin peut créer des schémas et des tables. Pour de plus amples informations, veuillez consulter Rôles définis par le système Amazon Redshift.

Étape 2 : Configuration des schémas

Au cours de cette étape, vous vous connectez à votre base de données en tant qu'administrateur de base de données. Ensuite, vous créez deux schémas et vous y ajoutez des données.

1. Connectez-vous à la base de données de développement en tant qu'utilisateur dbadmin à l'aide de l'éditeur de requêtes v2. Pour plus d'informations sur la connexion à une base de données, consultez la section Utilisation de l'éditeur de requêtes v2.
2. Pour créer les schémas de base de données des ventes et du marketing, utilisez l'exemple suivant.

```
CREATE SCHEMA sales;
CREATE SCHEMA marketing;
```

3. Pour créer et insérer des valeurs dans les tables du schéma de vente, utilisez l'exemple suivant.

```
CREATE TABLE sales.cat(
catid smallint,
catgroup varchar(10),
catname varchar(10),
catdesc varchar(50)
);
INSERT INTO sales.cat(SELECT * FROM category);

CREATE TABLE sales.dates(
dateid smallint,
caldate date,
day char(3),
week smallint,
month char(5),
qtr char(5),
year smallint,
holiday boolean
);
INSERT INTO sales.dates(SELECT * FROM date);
```

```
CREATE TABLE sales.events(
eventid integer,
venueid smallint,
catid smallint,
dateid smallint,
eventname varchar(200),
starttime timestamp
);
INSERT INTO sales.events(SELECT * FROM event);

 CREATE TABLE sales.sale(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp
);
INSERT INTO sales.sale(SELECT * FROM sales);
```

4. Pour créer et insérer des valeurs dans les tables du schéma marketing, utilisez l'exemple suivant.

```
CREATE TABLE marketing.cat(
catid smallint,
catgroup varchar(10),
catname varchar(10),
catdesc varchar(50)
);
INSERT INTO marketing.cat(SELECT * FROM category);

CREATE TABLE marketing.dates(
dateid smallint,
caldate date,
day char(3),
week smallint,
month char(5),
qtr char(5),
year smallint,
holiday boolean
);
```

```
INSERT INTO marketing.dates(SELECT * FROM date);

CREATE TABLE marketing.events(
eventid integer,
venueid smallint,
catid smallint,
dateid smallint,
eventname varchar(200),
starttime timestamp
);
INSERT INTO marketing.events(SELECT * FROM event);

CREATE TABLE marketing.sale(
marketingid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp
);
INSERT INTO marketing.sale(SELECT * FROM marketing);
```

Étape 3 : créer un utilisateur en lecture seule

Au cours de cette étape, vous créez un rôle en lecture seule et un utilisateur salesanalyst pour le rôle en lecture seule. L'analyste des ventes n'a besoin que d'un accès en lecture seule aux tables du schéma des ventes pour accomplir la tâche qui lui est assignée, à savoir trouver les événements qui ont généré les commissions les plus importantes.

1. Connectez-vous à la base de données en tant qu'utilisateur dbadmin.
2. Pour créer le rôle sales_ro, utilisez l'exemple suivant.

```
CREATE ROLE sales_ro;
```

3. Pour créer l'utilisateur salesanalyst, utilisez l'exemple suivant.

```
CREATE USER salesanalyst PASSWORD 'Test12345';
```

4. Pour autoriser l'utilisation du rôle sales_ro et sélectionner l'accès aux objets du schéma de vente, utilisez l'exemple suivant.

```
GRANT USAGE ON SCHEMA sales TO ROLE sales_ro;
GRANT SELECT ON ALL TABLES IN SCHEMA sales TO ROLE sales_ro;
```

5. Pour attribuer le rôle sales_ro à l'utilisateur salesanalyst, utilisez l'exemple suivant.

```
GRANT ROLE sales_ro TO salesanalyst;
```

Étape 4 : interroger les données en tant qu'utilisateur en lecture seule

Au cours de cette étape, l'utilisateur salesanalyst interroge les données du schéma de vente. L'utilisateur salesanalyst tente ensuite de mettre à jour une table et de lire des tables dans le schéma marketing.

1. Connectez-vous à la base de données en tant qu'utilisateur salesanalyst.
2. Pour trouver les 10 ventes avec les commissions les plus élevées, utilisez l'exemple suivant.

```
SET SEARCH_PATH TO sales;
SELECT DISTINCT events.dateid, sale.commission, cat.catname
FROM sale, events, dates, cat
WHERE events.dateid=dates.dateid AND events.dateid=sale.dateid AND events.catid =
 cat.catid
ORDER BY 2 DESC LIMIT 10;

+--------+------------+----------+
| dateid | commission | catname  |
+--------+------------+----------+
|   1880 |     1893.6 | Pop      |
|   1880 |     1893.6 | Opera    |
|   1880 |     1893.6 | Plays    |
|   1880 |     1893.6 | Musicals |
|   1861 |       1500 | Plays    |
|   2003 |       1500 | Pop      |
|   1861 |       1500 | Opera    |
|   2003 |       1500 | Plays    |
|   1861 |       1500 | Musicals |
```

```
|   1861 |       1500 | Pop      |
+--------+------------+----------+
```

3. Pour sélectionner 10 événements dans le tableau des événements du schéma de vente, utilisez l'exemple suivant.

```
SELECT * FROM sales.events LIMIT 10;

+---------+---------+-------+--------+--------------------+---------------------+
| eventid | venueid | catid | dateid |     eventname      |      starttime      |
+---------+---------+-------+--------+--------------------+---------------------+
|    4836 |      73 |     9 |   1871 | Soulfest           | 2008-02-14 19:30:00 |
|    5739 |      41 |     9 |   1871 | Fab Faux           | 2008-02-14 19:30:00 |
|     627 |     229 |     6 |   1872 | High Society       | 2008-02-15 14:00:00 |
|    2563 |     246 |     7 |   1872 | Hamlet             | 2008-02-15 20:00:00 |
|    7703 |      78 |     9 |   1872 | Feist              | 2008-02-15 14:00:00 |
|    7903 |      90 |     9 |   1872 | Little Big Town    | 2008-02-15 19:30:00 |
|    7925 |     101 |     9 |   1872 | Spoon              | 2008-02-15 19:00:00 |
|    8113 |      17 |     9 |   1872 | Santana            | 2008-02-15 15:00:00 |
|     463 |     303 |     8 |   1873 | Tristan und Isolde | 2008-02-16 19:00:00 |
|     613 |     236 |     6 |   1873 | Pal Joey           | 2008-02-16 15:00:00 |
+---------+---------+-------+--------+--------------------+---------------------+
```

4. Pour tenter de mettre à jour le nom de l'événement pour l'identifiant d'événement 1, exécutez l'exemple suivant. Cet exemple provoquera une erreur de refus d'autorisation car l'utilisateur salesanalyst ne dispose que des autorisations SELECT sur la table des événements du schéma de vente. Pour mettre à jour le tableau des événements, vous devez autoriser le rôle sales_ro à METTRE À JOUR. Pour plus d'informations sur l'octroi d'autorisations de mise à jour d'une table, consultez le paramètre UPDATE pourGRANT. Pour plus d'informations sur la commande UPDATE, consultezUPDATE.

```
UPDATE sales.events
SET eventname = 'Comment event'
WHERE eventid = 1;

ERROR: permission denied for relation events
```

5. Pour essayer de tout sélectionner dans le tableau des événements du schéma marketing, utilisez l'exemple suivant. Cet exemple provoquera une erreur de refus d'autorisation car l'utilisateur salesanalyst ne dispose que des autorisations SELECT pour la table des événements du schéma des ventes. Pour sélectionner des données dans le tableau des événements du schéma

marketing, vous devez accorder au rôle sales_ro les autorisations SELECT sur le tableau des événements du schéma marketing.

```
SELECT * FROM marketing.events;

               ERROR: permission denied for schema marketing
```

Étape 5 : Création d'un utilisateur en lecture-écriture

Au cours de cette étape, l'ingénieur commercial chargé de créer le pipeline d'extraction, de transformation et de chargement (ETL) pour le traitement des données dans le schéma de vente bénéficiera d'un accès en lecture seule, mais bénéficiera ultérieurement d'un accès en lecture et en écriture pour effectuer ses tâches.

1. Connectez-vous à la base de données en tant qu'utilisateur dbadmin.
2. Pour créer le rôle sales_rw dans le schéma de vente, utilisez l'exemple suivant.

```
CREATE ROLE sales_rw;
```

3. Pour créer l'utilisateur salesengineer, utilisez l'exemple suivant.

```
CREATE USER salesengineer PASSWORD 'Test12345';
```

4. Pour autoriser l'utilisation du rôle sales_rw et sélectionner l'accès aux objets du schéma de vente en lui attribuant le rôle sales_ro, utilisez l'exemple suivant. Pour plus d'informations sur la façon dont les rôles héritent des autorisations dans Amazon Redshift, consultez. Hiérarchie des rôles

```
GRANT ROLE sales_ro TO ROLE sales_rw;
```

5. Pour attribuer le rôle sales_rw à l'utilisateur salesengineer, utilisez l'exemple suivant.

```
GRANT ROLE sales_rw TO salesengineer;
```

Étape 6 : Interrogez les données en tant qu'utilisateur avec le rôle en lecture seule hérité

Au cours de cette étape, l'utilisateur salesengineer tente de mettre à jour le tableau des événements avant de recevoir des autorisations de lecture.

1. Connectez-vous à la base de données en tant qu'utilisateur salesengineer.
2. L'utilisateur salesengineer peut lire avec succès les données de la table des événements du schéma de vente. Pour sélectionner l'événement portant l'identifiant d'événement 1 dans le tableau des événements du schéma de vente, utilisez l'exemple suivant.

```
SELECT * FROM sales.events where eventid=1;

+---------+---------+-------+--------+-----------------+---------------------+
| eventid | venueid | catid | dateid |    eventname    |      starttime      |
+---------+---------+-------+--------+-----------------+---------------------+
|       1 |     305 |     8 |   1851 | Gotterdammerung | 2008-01-25 14:30:00 |
+---------+---------+-------+--------+-----------------+---------------------+
```

3. Pour essayer de tout sélectionner dans le tableau des événements du schéma marketing, utilisez l'exemple suivant. L'utilisateur salesengineer n'étant pas autorisé à accéder aux tables du schéma marketing, cette requête provoquera une erreur de refus d'autorisation. Pour sélectionner des données dans le tableau des événements du schéma marketing, vous devez accorder au rôle sales_rw les autorisations SELECT sur le tableau des événements du schéma marketing.

```
SELECT * FROM marketing.events;

ERROR: permission denied for schema marketing
```

4. Pour tenter de mettre à jour le nom de l'événement pour l'identifiant d'événement 1, exécutez l'exemple suivant. Cet exemple provoquera une erreur de refus d'autorisation car l'utilisateur salesengineer ne dispose que d'autorisations sélectionnées dans le tableau des événements du schéma de vente. Pour mettre à jour la table des événements, vous devez autoriser le rôle sales_rw à METTRE À JOUR.

```
UPDATE sales.events
SET eventname = 'Comment event'
WHERE eventid = 1;

ERROR: permission denied for relation events
```

Étape 7 : Accorder des autorisations de mise à jour et d'insertion pour le rôle de lecture-écriture

Au cours de cette étape, vous accordez des autorisations de mise à jour et d'insertion au rôle sales_rw.

1. Connectez-vous à la base de données en tant qu'utilisateur dbadmin.
2. Pour accorder les autorisations UPDATE, INSERT et DELETE au rôle sales_rw, utilisez l'exemple suivant.

```
GRANT UPDATE, INSERT, ON ALL TABLES IN SCHEMA sales TO role sales_rw;
```

Étape 8 : Interrogez les données en tant qu'utilisateur en lecture-écriture

Au cours de cette étape, l'ingénieur commercial met à jour le tableau avec succès une fois que son rôle a obtenu les autorisations d'insertion et de mise à jour. Ensuite, l'ingénieur commercial tente d'analyser et de vider le tableau des événements, mais n'y parvient pas.

1. Connectez-vous à la base de données en tant qu'utilisateur salesengineer.
2. Pour mettre à jour le nom de l'événement pour l'identifiant d'événement 1, exécutez l'exemple suivant.

```
UPDATE sales.events
SET eventname = 'Comment event'
WHERE eventid = 1;
```

3. Pour afficher la modification apportée dans la requête précédente, utilisez l'exemple suivant pour sélectionner l'événement portant l'identifiant d'événement 1 dans le tableau des événements du schéma de vente.

```
SELECT * FROM sales.events WHERE eventid=1;

+---------+---------+-------+--------+--------------+---------------------+
| eventid | venueid | catid | dateid |   eventname  |      starttime      |
+---------+---------+-------+--------+--------------+---------------------+
|       1 |     305 |     8 |   1851 | Comment event | 2008-01-25 14:30:00 |
+---------+---------+-------+--------+--------------+---------------------+
```

4. Pour analyser le tableau des événements mis à jour dans le schéma de vente, utilisez l'exemple suivant. Cet exemple provoquera une erreur de refus d'autorisation car l'utilisateur salesengineer ne dispose pas des autorisations nécessaires et n'est pas le propriétaire de la table des événements dans le schéma de vente. Pour analyser la table des événements, vous devez accorder au rôle sales_rw l'autorisation d'ANALYSER à l'aide de la commande GRANT. Pour plus d'informations sur la commande ANALYZE, consultezANALYSE.

```
ANALYZE sales.events;

               ERROR: skipping "events" --- only table or database owner can analyze
```

5. Pour vider le tableau des événements mis à jour, utilisez l'exemple suivant. Cet exemple provoquera une erreur de refus d'autorisation car l'utilisateur salesengineer ne dispose pas des autorisations nécessaires et n'est pas le propriétaire de la table des événements dans le schéma de vente. Pour vider la table des événements, vous devez accorder les autorisations du rôle sales_rw à VACUUM à l'aide de la commande GRANT. Pour plus d'informations sur la commande VACUUM, consultezVACUUM.

```
VACUUM sales.events;

ERROR: skipping "events" --- only table or database owner can vacuum it
```

Étape 9 : Analyser et vider les tables d'une base de données en tant qu'utilisateur administrateur

Au cours de cette étape, l'utilisateur de dbadmin analyse et vide toutes les tables. L'utilisateur dispose d'autorisations d'administrateur sur cette base de données, ce qui lui permet d'exécuter ces commandes.

1. Connectez-vous à la base de données en tant qu'utilisateur dbadmin.
2. Pour analyser le tableau des événements dans le schéma des ventes, utilisez l'exemple suivant.

```
ANALYZE sales.events;
```

3. Pour vider le tableau des événements dans le schéma de vente, utilisez l'exemple suivant.

```
VACUUM sales.events;
```

4. Pour analyser le tableau des événements dans le schéma marketing, utilisez l'exemple suivant.

```
ANALYZE marketing.events;
```

5. Pour supprimer le tableau des événements dans le schéma marketing, utilisez l'exemple suivant.

```
VACUUM marketing.events;
```

Étape 10 : tronquer les tables en tant qu'utilisateur en lecture-écriture

Au cours de cette étape, l'utilisateur salesengineer tente de tronquer le tableau des événements dans le schéma de vente, mais n'y parvient que lorsque l'utilisateur dbadmin lui accorde des autorisations de troncature.

1. Connectez-vous à la base de données en tant qu'utilisateur salesengineer.
2. Pour essayer de supprimer toutes les lignes de la table des événements dans le schéma de vente, utilisez l'exemple suivant. Cet exemple provoquera une erreur car l'utilisateur salesengineer ne dispose pas des autorisations nécessaires et n'est pas le propriétaire de la table des événements dans le schéma de vente. Pour tronquer la table des événements, vous devez accorder au rôle sales_rw l'autorisation de TRUNCATE à l'aide de la commande GRANT. Pour plus d'informations sur la commande TRUNCATE, consultez TRUNCATE.

```
TRUNCATE sales.events;

ERROR: must be owner of relation events
```

3. Connectez-vous à la base de données en tant qu'utilisateur dbadmin.
4. Pour accorder des privilèges de troncature de table au rôle sales_rw, utilisez l'exemple suivant.

```
GRANT TRUNCATE TABLE TO role sales_rw;
```

5. Connectez-vous à la base de données en tant qu'utilisateur salesengineer à l'aide de l'éditeur de requêtes v2.
6. Pour lire les 10 premiers événements du tableau des événements du schéma de vente, utilisez l'exemple suivant.

```
SELECT * FROM sales.events ORDER BY eventid LIMIT 10;
```

```
+---------+---------+-------+--------+-----------------------------
+---------------------+
| eventid | venueid | catid | dateid |          eventname          |      starttime
    |
+---------+---------+-------+--------+-----------------------------
+---------------------+
|       1 |     305 |     8 |   1851 | Comment event               | 2008-01-25
 14:30:00 |
|       2 |     306 |     8 |   2114 | Boris Godunov               | 2008-10-15
 20:00:00 |
|       3 |     302 |     8 |   1935 | Salome                      | 2008-04-19
 14:30:00 |
|       4 |     309 |     8 |   2090 | La Cenerentola (Cinderella) | 2008-09-21
 14:30:00 |
|       5 |     302 |     8 |   1982 | Il Trovatore                | 2008-06-05
 19:00:00 |
|       6 |     308 |     8 |   2109 | L Elisir d Amore            | 2008-10-10
 19:30:00 |
|       7 |     309 |     8 |   1891 | Doctor Atomic               | 2008-03-06
 14:00:00 |
|       8 |     302 |     8 |   1832 | The Magic Flute             | 2008-01-06
 20:00:00 |
|       9 |     308 |     8 |   2087 | The Fly                     | 2008-09-18
 19:30:00 |
|      10 |     305 |     8 |   2079 | Rigoletto                   | 2008-09-10
 15:00:00 |
+---------+---------+-------+--------+-----------------------------
+---------------------+
```

7. Pour tronquer le tableau des événements dans le schéma de vente, utilisez l'exemple suivant.

```
TRUNCATE sales.events;
```

8. Pour lire les données de la table des événements mise à jour dans le schéma de vente, utilisez l'exemple suivant.

```
SELECT * FROM sales.events ORDER BY eventid LIMIT 10;

+---------+---------+-------+--------+-----------------------------
+---------------------+
| eventid | venueid | catid | dateid |          eventname          |      starttime
    |
```

```
+---------+---------+-------+--------+----------------------------
+---------------------+
```

Créez des rôles en lecture seule et en lecture-écriture pour le schéma marketing (facultatif)

Au cours de cette étape, vous créez des rôles en lecture seule et en lecture-écriture pour le schéma marketing.

1. Connectez-vous à la base de données en tant qu'utilisateur dbadmin.
2. Pour créer des rôles en lecture seule et en lecture-écriture pour le schéma marketing, utilisez l'exemple suivant.

```
CREATE ROLE marketing_ro;

CREATE ROLE marketing_rw;

GRANT USAGE ON SCHEMA marketing TO ROLE marketing_ro, ROLE marketing_rw;

GRANT SELECT ON ALL TABLES IN SCHEMA marketing TO ROLE marketing_ro;

GRANT ROLE marketing_ro TO ROLE marketing_rw;

GRANT INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA marketing TO ROLE marketing_rw;

CREATE USER marketinganalyst PASSWORD 'Test12345';

CREATE USER marketingengineer PASSWORD 'Test12345';

GRANT ROLE marketing_ro TO marketinganalyst;

GRANT ROLE marketing_rw TO marketingengineer;
```

Fonctions du système pour RBAC (en option)

Amazon Redshift dispose de deux fonctions pour fournir des informations système sur l'appartenance des utilisateurs et des rôles à des groupes ou rôles supplémentaires : role_is_member_of et user_is_member_of. Ces fonctions sont disponibles pour les superutilisateurs et les utilisateurs réguliers. Les superutilisateurs peuvent vérifier toutes les appartenances aux rôles. Les utilisateurs réguliers ne peuvent vérifier l'adhésion que pour les rôles auxquels ils ont été autorisés à accéder.

Pour utiliser la fonction role_is_member_of

1. Connectez-vous à la base de données en tant qu'utilisateur salesengineer.
2. Pour vérifier si le rôle sales_rw est membre du rôle sales_ro, utilisez l'exemple suivant.

```
SELECT role_is_member_of('sales_rw', 'sales_ro');

+-------------------+
| role_is_member_of |
+-------------------+
| true              |
+-------------------+
```

3. Pour vérifier si le rôle sales_ro est membre du rôle sales_rw, utilisez l'exemple suivant.

```
SELECT role_is_member_of('sales_ro', 'sales_rw');

+-------------------+
| role_is_member_of |
+-------------------+
| false             |
+-------------------+
```

Pour utiliser la fonction user_is_member_of

1. Connectez-vous à la base de données en tant qu'utilisateur salesengineer.
2. L'exemple suivant tente de vérifier l'appartenance de l'utilisateur salesanalyst. Cette requête génère une erreur car salesengineer n'a pas accès à salesanalyst. Pour exécuter correctement cette commande, connectez-vous à la base de données en tant qu'utilisateur salesanalyst et utilisez l'exemple.

```
SELECT user_is_member_of('salesanalyst', 'sales_ro');

ERROR
```

3. Connectez-vous à la base de données en tant que superutilisateur.
4. Pour vérifier l'appartenance de l'utilisateur salesanalyst lorsqu'il est connecté en tant que superutilisateur, utilisez l'exemple suivant.

```
SELECT user_is_member_of('salesanalyst', 'sales_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| true              |
+-------------------+
```

5. Connectez-vous à la base de données en tant qu'utilisateur dbadmin.
6. Pour vérifier l'adhésion de l'utilisateur salesengineer, utilisez l'exemple suivant.

```
SELECT user_is_member_of('salesengineer', 'sales_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| true              |
+-------------------+

SELECT user_is_member_of('salesengineer', 'marketing_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| false             |
+-------------------+

SELECT user_is_member_of('marketinganalyst', 'sales_ro');

+-------------------+
| user_is_member_of |
+-------------------+
| false             |
+-------------------+
```

Vues du système pour RBAC (en option)

Pour consulter les rôles, l'attribution des rôles aux utilisateurs, la hiérarchie des rôles et les privilèges pour les objets de base de données via des rôles, utilisez les vues système d'Amazon Redshift. Ces vues sont accessibles aux superutilisateurs et aux utilisateurs réguliers. Les superutilisateurs peuvent

vérifier tous les détails des rôles. Les utilisateurs réguliers peuvent uniquement vérifier les détails des rôles auxquels ils ont été autorisés à accéder.

1. Pour afficher la liste des utilisateurs auxquels des rôles ont été explicitement attribués dans le cluster, utilisez l'exemple suivant.

```
SELECT * FROM svv_user_grants;
```

2. Pour afficher la liste des rôles auxquels des rôles sont explicitement attribués dans le cluster, utilisez l'exemple suivant.

```
SELECT * FROM svv_role_grants;
```

Pour obtenir la liste complète des vues du système, reportez-vous àVues de métadonnées SVV.

Utiliser la sécurité au niveau des lignes avec le RBAC (facultatif)

Pour contrôler l'accès granulaire à vos données sensibles, utilisez la sécurité au niveau des lignes (RLS). Pour plus d'informations sur RLS, consultez Sécurité au niveau des lignes.

Dans cette section, vous créez une politique RLS qui autorise l'`salesengineer`utilisateur à afficher uniquement les lignes du `cat` tableau qui ont la `catdesc` valeur de Major League Baseball. Vous interrogez ensuite la base de données en tant qu'`salesengineer`utilisateur.

1. Connectez-vous à la base de données en tant qu'`salesengineer`utilisateur.
2. Pour afficher les 5 premières entrées du `cat` tableau, utilisez l'exemple suivant.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
|     1 | Sports   | MLB     | Major League Baseball           |
|     2 | Sports   | NHL     | National Hockey League          |
|     3 | Sports   | NFL     | National Football League        |
|     4 | Sports   | NBA     | National Basketball Association |
|     5 | Sports   | MLS     | Major League Soccer             |
```

```
+-------+----------+---------+--------------------------------+
```

3. Connectez-vous à la base de données en tant qu'`dbadmin`utilisateur.
4. Pour créer une politique RLS pour la `catdesc` colonne du `cat` tableau, utilisez l'exemple suivant.

```
CREATE RLS POLICY policy_mlb_engineer
WITH (catdesc VARCHAR(50))
USING (catdesc = 'Major League Baseball');
```

5. Pour associer la politique RLS au `sales_rw` rôle, utilisez l'exemple suivant.

```
ATTACH RLS POLICY policy_mlb_engineer ON sales.cat TO ROLE sales_rw;
```

6. Pour modifier le tableau afin d'activer le protocole RLS, utilisez l'exemple suivant.

```
ALTER TABLE sales.cat ROW LEVEL SECURITY ON;
```

7. Connectez-vous à la base de données en tant qu'`salesengineer`utilisateur.
8. Pour essayer d'afficher les 5 premières entrées du `cat` tableau, utilisez l'exemple suivant. Notez que seules les entrées apparaissent uniquement lorsque la `catdesc` colonne est`Major League Baseball`.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+----------------------+
| catid | catgroup | catname |        catdesc       |
+-------+----------+---------+----------------------+
|     1 | Sports   | MLB     | Major League Baseball |
+-------+----------+---------+----------------------+
```

9. Connectez-vous à la base de données en tant qu'`salesanalyst`utilisateur.
10. Pour essayer d'afficher les 5 premières entrées du `cat` tableau, utilisez l'exemple suivant. Notez qu'aucune entrée n'apparaît car la politique par défaut de refuser tout est appliquée.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;
```

```
+-------+----------+---------+----------------------+
| catid | catgroup | catname |        catdesc       |
+-------+----------+---------+----------------------+
```

11. Connectez-vous à la base de données en tant qu'`dbadmin`utilisateur.
12. Pour accorder l'autorisation IGNORE RLS au `sales_ro` rôle, utilisez l'exemple suivant. Cela donne à l'`salesanalyst`utilisateur l'autorisation d'ignorer les politiques RLS puisqu'il est membre du `sales_ro` rôle.

```
GRANT IGNORE RLS TO ROLE sales_ro;
```

13. Connectez-vous à la base de données en tant qu'`salesanalyst`utilisateur.
14. Pour afficher les 5 premières entrées du `cat` tableau, utilisez l'exemple suivant.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
|     1 | Sports   | MLB     | Major League Baseball           |
|     2 | Sports   | NHL     | National Hockey League          |
|     3 | Sports   | NFL     | National Football League        |
|     4 | Sports   | NBA     | National Basketball Association |
|     5 | Sports   | MLS     | Major League Soccer             |
+-------+----------+---------+---------------------------------+
```

15. Connectez-vous à la base de données en tant qu'`dbadmin`utilisateur.
16. Pour révoquer l'autorisation IGNORE RLS du `sales_ro` rôle, utilisez l'exemple suivant.

```
REVOKE IGNORE RLS FROM ROLE sales_ro;
```

17. Connectez-vous à la base de données en tant qu'`salesanalyst`utilisateur.
18. Pour essayer d'afficher les 5 premières entrées du `cat` tableau, utilisez l'exemple suivant. Notez qu'aucune entrée n'apparaît car la politique par défaut de refuser tout est appliquée.

```
SELECT *
FROM sales.cat
```

```
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+----------------------+
| catid | catgroup | catname |        catdesc       |
+-------+----------+---------+----------------------+
```

19. Connectez-vous à la base de données en tant qu'`dbadmin`utilisateur.
20. Pour détacher la politique RLS de la `cat` table, utilisez l'exemple suivant.

```
DETACH RLS POLICY policy_mlb_engineer ON cat FROM ROLE sales_rw;
```

21. Connectez-vous à la base de données en tant qu'`salesanalyst`utilisateur.
22. Pour essayer d'afficher les 5 premières entrées du `cat` tableau, utilisez l'exemple suivant. Notez qu'aucune entrée n'apparaît car la politique par défaut de refuser tout est appliquée.

```
SELECT *
FROM sales.cat
ORDER BY catid ASC
LIMIT 5;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
|     1 | Sports   | MLB     | Major League Baseball           |
|     2 | Sports   | NHL     | National Hockey League          |
|     3 | Sports   | NFL     | National Football League        |
|     4 | Sports   | NBA     | National Basketball Association |
|     5 | Sports   | MLS     | Major League Soccer             |
+-------+----------+---------+---------------------------------+
```

23. Connectez-vous à la base de données en tant qu'`dbadmin`utilisateur.
24. Pour supprimer la politique RLS, utilisez l'exemple suivant.

```
DROP RLS POLICY policy_mlb_engineer;
```

25. Pour supprimer RLS, utilisez l'exemple suivant.

```
ALTER TABLE cat ROW LEVEL SECURITY OFF;
```

Rubriques en relation

Pour plus d'informations sur le RBAC, consultez la documentation suivante :

- Hiérarchie des rôles
- Attribution des rôles
- Autorisations d’objet de base de données
- Instruction ALTER DEFAULT PRIVILEGES pour le RBAC

Sécurité au niveau des lignes

Grâce à la sécurité au niveau des lignes (RLS) d’Amazon Redshift, vous pouvez bénéficier d’un contrôle précis des accès à vos données sensibles. Vous pouvez décider quels utilisateurs ou rôles peuvent accéder à des enregistrements de données spécifiques au sein de schémas ou de tables, en fonction des politiques de sécurité définies au niveau des objets de base de données. En plus de la sécurité au niveau des colonnes, où vous pouvez accorder aux utilisateurs des autorisations pour un sous-ensemble de colonnes, utilisez des politiques RLS pour restreindre davantage l’accès à certaines lignes des colonnes visibles. Pour plus d’informations sur la sécurité au niveau des colonnes, consultez Notes d’utilisation pour le contrôle d’accès de niveau colonne.

Lorsque vous appliquez des politiques RLS sur des tables, vous pouvez restreindre les ensembles de résultats renvoyés lorsque les utilisateurs exécutent des requêtes.

Lorsque vous créez des politiques RLS, vous pouvez spécifier des expressions qui déterminent si Amazon Redshift renvoie toutes lignes existantes dans une table dans une requête. En créant des politiques RLS pour limiter l’accès, vous n’avez pas besoin d’ajouter ou d’externaliser des conditions supplémentaires dans vos requêtes.

Lorsque vous créez des politiques RLS, nous vous recommandons de créer des politiques simples et d’éviter les instructions complexes dans les politiques. Lorsque vous définissez des politiques RLS, n’utilisez pas trop de jointures de tables basées sur des politiques dans la définition de stratégie.

Lorsqu’une politique fait référence à une table de recherche, Amazon Redshift analyse la table supplémentaire en plus de la table sur laquelle la politique existe. Il y aura des différences de performances entre la même requête pour un utilisateur auquel une politique RLS est attachée et un utilisateur sans politique attachée.

Utilisation des politiques RLS dans les instructions SQL

Lorsque vous utilisez des politiques RLS dans des instructions SQL, Amazon Redshift applique les règles suivantes :

- Amazon Redshift applique par défaut les politiques RLS aux instructions SELECT, UPDATE et DELETE.
- Pour SELECT et UNLOAD, Amazon Redshift filtre les lignes en fonction de la politique que vous avez définie.
- Pour UPDATE, Amazon Redshift ne met à jour que les lignes que vous voyez. Si une politique restreint un sous-ensemble des lignes d'une table, vous ne pouvez pas les mettre à jour.
- Pour DELETE, vous ne pouvez supprimer que les lignes que vous voyez. Si une politique restreint un sous-ensemble des lignes d'une table, vous ne pouvez pas les supprimer. Pour TRUNCATE, vous pouvez toujours tronquer la table.
- Pour CREATE TABLE LIKE, les tables créées avec les options LIKE n'héritent pas des paramètres d'autorisation de la table source. De même, la table cible n'hérite pas des politiques RLS de la table source.

Association de plusieurs politiques par utilisateur

Dans Amazon Redshift, RLS prend en charge l'association de plusieurs politiques par utilisateur et par objet. Lorsque plusieurs politiques sont définies pour un utilisateur, Amazon Redshift applique toutes les politiques avec la syntaxe AND ou OR en fonction du paramètre RLS CONJUNCTION TYPE défini pour la table. Pour plus d'informations sur les types de conjonction, consultez ALTER TABLE.

Plusieurs politiques d'une même table peuvent vous être associées. Soit plusieurs politiques vous sont directement attachées, soit vous appartenez à plusieurs rôles, et les rôles ont différentes politiques qui leur sont attachées.

Lorsque les différentes politiques doivent restreindre l'accès aux lignes dans une relation donnée, vous pouvez définir l'élément RLS CONJUNCTION TYPE de la relation à AND. Prenez l'exemple de code suivant. Alice ne peut voir que les événements sportifs dont le « catname » est NBA, conformément à la politique spécifiée.

```
-- Create an analyst role and grant it to a user named Alice.
```

```
CREATE ROLE analyst;
CREATE USER alice WITH PASSWORD 'Name_is_alice_1';
GRANT ROLE analyst TO alice;

-- Create an RLS policy that only lets the user see sports.
CREATE RLS POLICY policy_sports
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Sports');

-- Create an RLS policy that only lets the user see NBA.
CREATE RLS POLICY policy_nba
WITH (catname VARCHAR(10))
USING (catname = 'NBA');

-- Attach both to the analyst role.
ATTACH RLS POLICY policy_sports ON category TO ROLE analyst;
ATTACH RLS POLICY policy_nba ON category TO ROLE analyst;

-- Activate RLS on the category table with AND CONJUNCTION TYPE.
ALTER TABLE category ROW LEVEL SECURITY ON CONJUNCTION TYPE AND;

-- Change session to Alice.
SET SESSION AUTHORIZATION alice;

-- Select all from the category table.
SELECT catgroup, catname
FROM category;

 catgroup | catname
---------+---------
 Sports   | NBA
(1 row)
```

Lorsque les différentes politiques doivent permettre aux utilisateurs de voir davantage de lignes dans une relation donnée, l'utilisateur peut définir l'élément RLS CONJUNCTION TYPE de la relation à OR. Prenez l'exemple de code suivant. Alice ne peut voir que « Concerts » et « Sports » conformément à la politique spécifiée.

```
-- Create an analyst role and grant it to a user named Alice.
CREATE ROLE analyst;
CREATE USER alice WITH PASSWORD 'Name_is_alice_1';
GRANT ROLE analyst TO alice;
```

```
-- Create an RLS policy that only lets the user see concerts.
CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Concerts');

-- Create an RLS policy that only lets the user see sports.
CREATE RLS POLICY policy_sports
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Sports');

-- Attach both to the analyst role.
ATTACH RLS POLICY policy_concerts ON category TO ROLE analyst;
ATTACH RLS POLICY policy_sports ON category TO ROLE analyst;

-- Activate RLS on the category table with OR CONJUNCTION TYPE.
ALTER TABLE category ROW LEVEL SECURITY ON CONJUNCTION TYPE OR;

-- Change session to Alice.
SET SESSION AUTHORIZATION alice;

-- Select all from the category table.
SELECT catgroup, count(*)
FROM category
GROUP BY catgroup ORDER BY catgroup;

 catgroup | count
---------+-------
 Concerts |  3
 Sports   |  5
(2 rows)
```

Propriété et gestion des politiques RLS

En tant que superutilisateur, administrateur de sécurité ou utilisateur doté du rôle sys:secadmin, vous pouvez créer, modifier, joindre et détacher des politiques RLS. Les politiques RLS peuvent être attachées à des tables, à des vues, à des vues à liaison tardive (LBVs) et à des vues matérialisées (MVs). Au niveau de l’objet, vous pouvez activer ou désactiver la sécurité au niveau des lignes sans modifier la définition du schéma pour les tables.

Pour commencer à utiliser la sécurité au niveau des lignes, vous pouvez utiliser les instructions SQL suivantes :

- Utilisez l'instruction ALTER TABLE pour activer ou désactiver le protocole RLS sur une table, une vue ou une vue à reliure tardive. Pour de plus amples informations, veuillez consulter ALTER TABLE.
- Utilisez l'instruction ALTER MATERIALIZED VIEW to pour activer ou désactiver le RLS sur une vue matérialisée (MV). Pour de plus amples informations, veuillez consulter ALTER MATERIALIZED VIEW.
- Utilisez l’instruction CREATE RLS POLICY pour créer une politique de sécurité pour une ou plusieurs tables et spécifier un ou plusieurs utilisateurs ou rôles dans la stratégie.

  Pour de plus amples informations, veuillez consulter CREATE RLS POLICY.
- Utilisez l’instruction ALTER RLS POLICY pour modifier la politique, par exemple en modifiant la définition de la politique. Vous pouvez utiliser la même politique pour plusieurs tables ou vues.

  Pour de plus amples informations, veuillez consulter ALTER RLS POLICY.
- Utilisez l’instruction ATTACH RLS POLICY pour attacher une politique à une ou plusieurs relations, à un ou plusieurs utilisateurs, ou à des rôles.

  Pour de plus amples informations, veuillez consulter ATTACH RLS POLICY.
- Utilisez l'instruction DETACH RLS POLICY pour détacher une politique d'une ou de plusieurs relations, d'un ou de plusieurs utilisateurs ou de rôles.

  Pour de plus amples informations, veuillez consulter DETACH RLS POLICY.
- Utilisez l’instruction DROP RLS POLICY pour supprimer une stratégie.

  Pour de plus amples informations, veuillez consulter DROP RLS POLICY.
- Utilisez les instructions GRANT et REVOKE pour accorder et révoquer explicitement les autorisations SELECT aux politiques RLS qui font référence à des tables de recherche. Pour plus d’informations, consultez GRANT et REVOKE.

Pour surveiller les politiques créées, les utilisateurs disposant du rôle sys:secadmin peuvent consulter SVV_RLS_POLICY et SVV_RLS_ATTACHED_POLICY.

Pour répertorier les relations protégées par RLS, sys:secadmin peut afficher. SVV_RLS_RELATION

Pour suivre l’application des politiques RLS sur les requêtes qui font référence à des relations protégées par RLS, un super-utilisateur, un utilisateur disposant du rôle sys:operator ou tout autre utilisateur disposant de l’autorisation système ACCESS SYSTEM TABLE peut consulter

SVV_RLS_APPLIED_POLICY. Remarquez que les utilisateurs disposant du rôle sys:secadmin ne disposent pas de ces autorisations par défaut.

Pour permettre aux utilisateurs d'accéder pleinement à une relation protégée par RLS, vous pouvez accorder l'autorisation IGNORE RLS. Les superutilisateurs ou sys:secadmin reçoivent automatiquement le protocole IGNORE RLS. Pour de plus amples informations, veuillez consulter GRANT.

Pour expliquer les filtres de politique RLS d'une requête dans le plan EXPLAIN afin de dépanner les requêtes liées à RLS, vous pouvez accorder l'autorisation EXPLAIN RLS à n'importe quel utilisateur. Pour plus d'informations, consultez GRANT et EXPLAIN.

Objets et principes dépendants des politiques

Afin de garantir la sécurité de toutes les applications et pour éviter que les objets de politique ne deviennent obsolètes ou non valides, Amazon Redshift ne permet pas de supprimer ou de modifier des objets référencés par les politiques RLS.

La liste suivante répertorie les dépendances d'objets de schéma qu'Amazon Redshift suit pour les politiques RLS.

- Lors du suivi de la dépendance des objets de schéma pour la table cible, Amazon Redshift suit les règles suivantes :
  - Amazon Redshift détache la politique d'une relation, d'un utilisateur, d'un rôle ou d'un public lorsque vous supprimez une table cible.
  - Lorsque vous modifiez le nom d'une table cible, cela n'a aucun impact sur les politiques attachées.
  - Vous ne pouvez supprimer les colonnes de la table cible référencée dans la définition de la politique que si vous supprimez ou détachez d'abord la stratégie. Cela s'applique également lorsque l'option CASCADE est spécifiée. Vous pouvez supprimer d'autres colonnes dans la table cible.
  - Vous ne pouvez pas renommer les colonnes référencées de la table cible. Pour renommer des colonnes référencées, détachez d'abord la stratégie. Cela s'applique également lorsque l'option CASCADE est spécifiée.
  - Vous ne pouvez pas modifier le type de la colonne référencée, même lorsque vous spécifiez l'option CASCADE.
- Lors du suivi de la dépendance des objets de schéma pour la table de recherche, Amazon Redshift suit les règles suivantes :

  - Vous ne pouvez pas supprimer une table de recherche. Pour supprimer une table de recherche, supprimez d'abord la politique dans laquelle elle est référencée.
  - Vous ne pouvez pas renommer une table de recherche. Pour renommer une table de recherche, supprimez d'abord la politique dans laquelle elle est référencée. Cela s'applique également lorsque l'option CASCADE est spécifiée.
  - Vous ne pouvez pas supprimer les colonnes de la table de recherche utilisées dans la définition de la stratégie. Pour supprimer les colonnes de la table de recherche utilisées dans la définition de la stratégie, supprimez d'abord la politique dans laquelle la table de recherche est référencée. Cela s'applique également lorsque l'option CASCADE est spécifiée dans l'instruction ALTER TABLE DROP COLUMN. Vous pouvez supprimer d'autres colonnes dans la table de recherche.
  - Vous ne pouvez pas renommer les colonnes référencées de la table de recherche. Pour renommer des colonnes référencées, supprimez d'abord la politique dans laquelle la table de recherche est référencée. Cela s'applique également lorsque l'option CASCADE est spécifiée.
  - Vous ne pouvez pas modifier le type de la colonne référencée.
- Lorsqu'un utilisateur ou un rôle est supprimé, Amazon Redshift détache automatiquement toutes les politiques attachées à l'utilisateur ou au rôle.
- Lorsque vous utilisez l'option CASCADE dans l'instruction DROP SCHEMA, Amazon Redshift supprime également les relations dans le schéma. Amazon Redshift supprime également les relations dans tous les autres schémas qui dépendent des relations du schéma supprimé. Pour une relation qui est une table de recherche dans une stratégie, Amazon Redshift ne parvient pas à exécuter l'instruction DROP SCHEMA DDL. Pour toutes les relations supprimées par l'instruction DROP SCHEMA, Amazon Redshift détache toutes les politiques attachées à ces relations.
- Vous ne pouvez supprimer une fonction de recherche (une fonction qui est référencée dans une définition de stratégie) que si vous supprimez également la stratégie. Cela s'applique également lorsque l'option CASCADE est spécifiée.
- Lorsqu'une politique est attachée à une table, Amazon Redshift vérifie si cette table est une table de recherche dans une autre stratégie. Si c'est le cas, Amazon Redshift ne permet pas d'attacher une politique à cette table.
- Lors de la création d'une politique RLS, Amazon Redshift vérifie si cette table est une table cible pour une autre politique RLS. Si c'est le cas, Amazon Redshift ne permet pas de créer une politique sur cette table.

exemple

L'exemple suivant illustre le suivi de la dépendance des schémas.

```
-- The CREATE and ATTACH policy statements for `policy_events` references some
-- target and lookup tables.
-- Target tables are tickit_event_redshift and target_schema.target_event_table.
-- Lookup table is tickit_sales_redshift.
-- Policy `policy_events` has following dependencies:
--   table tickit_sales_redshift column eventid, qtysold
--   table tickit_event_redshift column eventid
--   table target_event_table column eventid
--   schema public and target_schema
CREATE RLS POLICY policy_events
WITH (eventid INTEGER)
USING (
    eventid IN (SELECT eventid FROM tickit_sales_redshift WHERE qtysold <3)
);

ATTACH RLS POLICY policy_events ON tickit_event_redshift TO ROLE analyst;

ATTACH RLS POLICY policy_events ON target_schema.target_event_table TO ROLE consumer;
```

Considérations et limites relatives à l'utilisation des politiques RLS

Considérations

Voici des éléments à prendre en compte pour travailler avec des politiques RLS :

- Amazon Redshift applique les politiques RLS aux instructions SELECT, UPDATE ou DELETE.
- Amazon Redshift n’applique pas les politiques RLS aux instructions INSERT, COPY, ALTER TABLE APPEND.
- Les politiques RLS peuvent être attachées à des tables, à des vues, à des vues à liaison tardive (LBVs) et à des vues matérialisées (MVs).
- La sécurité au niveau des lignes fonctionne avec la sécurité au niveau des colonnes pour protéger vos données.
- Lorsque RLS est activé pour la relation source, Amazon Redshift prend en charge l’instruction ALTER TABLE APPEND pour les super-utilisateurs, les utilisateurs auxquels le privilège système IGNORE RLS a été explicitement accordé ou le rôle sys:secadmin. Dans ce cas, vous pouvez exécuter l’instruction ALTER TABLE APPEND pour ajouter des lignes à une table cible en

déplaçant les données à partir d'une table source existante. Amazon Redshift déplace tous les tuples de la relation source vers la relation cible. L'état RLS de la relation cible n'affecte pas l'instruction ALTER TABLE APPEND.

- Pour faciliter la migration à partir d'autres systèmes d'entrepôts des données, vous pouvez définir et récupérer des variables de contexte de session personnalisées pour une connexion en spécifiant le nom et la valeur de la variable.

  L'exemple suivant définit les variables de contexte de session pour une politique de sécurité au niveau des lignes (RLS).

```
-- Set a customized context variable.
SELECT set_config('app.category', 'Concerts', FALSE);

-- Create a RLS policy using current_setting() to get the value of a customized
 context variable.
CREATE RLS POLICY policy_categories
WITH (catgroup VARCHAR(10))
USING (catgroup = current_setting('app.category', FALSE));

-- Set correct roles and attach the policy on the target table to one or more roles.
ATTACH RLS POLICY policy_categories ON tickit_category_redshift TO ROLE analyst, ROLE
 dbadmin;
```

  Pour plus de détails sur la façon de définir et de récupérer des variables de contexte de session personnalisées SETSET_CONFIG, reportez-vous àMONTRER,CURRENT_SETTING, etRESET. Pour plus d'informations sur la modification de la configuration du serveur en général, rendez-vous surModification de la configuration du serveur.

  > **Important**
  >
  > Lorsque vous utilisez des variables de contexte de session dans des politiques RLS, la politique de sécurité dépend de l'utilisateur ou du rôle qui invoque la politique. Veillez à éviter les failles de sécurité lorsque vous utilisez des variables de contexte de session dans les politiques RLS.

- Le changement d'utilisateur de session à l'aide de SET SESSION AUTHORIZATION entre DECLARE et FETCH ou entre les instructions FETCH suivantes n'actualisera pas le plan déjà préparé en fonction des politiques utilisateur au moment de DECLARE. Évitez de changer d'utilisateur de session lorsque des curseurs sont utilisés avec des tables protégées par RLS.

- Lorsque les objets de base contenus dans un objet de vue sont protégés par RLS, les politiques associées à l'utilisateur exécutant la requête sont appliquées aux objets de base respectifs. Ceci est différent des contrôles d'autorisation au niveau des objets, où les autorisations du propriétaire de la vue sont comparées aux objets de base de la vue. Vous pouvez consulter les relations protégées par RLS d'une requête dans sa sortie du plan EXPLAIN.
- Lorsqu'une fonction définie par l'utilisateur (UDF) est référencée dans une politique RLS d'une relation attachée à un utilisateur, l'utilisateur doit disposer de l'autorisation EXECUTE sur la fonction UDF pour interroger la relation.
- La sécurité au niveau des lignes peut limiter l'optimisation des requêtes. Nous vous recommandons d'évaluer soigneusement les performances des requêtes avant de déployer des vues protégées par RLS sur de grands jeux de données.
- Les politiques de sécurité au niveau des lignes appliquées aux vues à liaison tardive peuvent être intégrées dans des tables fédérées. Ces politiques RLS peuvent être visibles dans les journaux des moteurs de traitement externes.

Limites

Voici les limites lorsque vous travaillez avec des politiques RLS :

- Les politiques RLS ne peuvent pas être attachées à des tables externes et à plusieurs autres types de relations. Pour de plus amples informations, veuillez consulter ATTACH RLS POLICY.
- Amazon Redshift prend en charge les instructions SELECT pour certaines politiques RLS avec des recherches comportant des jointures complexes, mais ne prend pas en charge les instructions UPDATE ou DELETE. Dans les cas où des instructions UPDATE ou DELETE sont utilisées, Amazon Redshift renvoie le message d'erreur suivant :

```
ERROR: One of the RLS policies on target relation is not supported in UPDATE/DELETE.
```

- Chaque fois qu'une fonction UDF est référencée dans une politique RLS d'une relation attachée à un utilisateur, l'utilisateur doit disposer de l'autorisation EXECUTE sur la fonction UDF pour interroger la relation.
- Les sous-requêtes corrélées ne sont pas prises en charge. Amazon Redshift renvoie l'erreur suivante :

```
ERROR: RLS policy could not be rewritten.
```

- Amazon Redshift ne prend pas en charge l'unité de partage des données avec RLS. Si, dans le cadre d'une relation, RLS n'est pas désactivé pour les unités de partage des données, la requête échoue sur le cluster consommateur avec l'erreur suivante :

  ```
  RLS-protected relation "rls_protected_table" cannot be accessed via datasharing
   query.
  ```

  Vous pouvez désactiver le protocole RLS pour les partages de données à l'aide de la commande ALTER TABLE avec le paramètre ROW LEVEL SECURITY OFF FOR DATASHARES. Pour plus d'informations sur l'utilisation d'ALTER TABLE pour activer ou désactiver RLS, consultez. ALTER TABLE

- Dans les requêtes entre bases de données, Amazon Redshift bloque les lectures vers des relations protégées par RLS. Les utilisateurs disposant de l'autorisation IGNORE RLS peuvent accéder à la relation protégée à l'aide de requêtes entre bases de données. Lorsqu'un utilisateur ne disposant pas de l'autorisation IGNORE RLS accède à une relation protégée par RLS via une requête entre bases de données, l'erreur suivante s'affiche :

  ```
  RLS-protected relation "rls_protected_table" cannot be accessed via cross-database
   query.
  ```

- ALTER RLS POLICY prend uniquement en charge la modification d'une politique RLS à l'aide de la clause USING (using_predicate_exp). Vous ne pouvez pas modifier une politique RLS à l'aide d'une clause WITH lorsque vous exécutez ALTER RLS POLICY.
- Vous ne pouvez pas interroger les relations dont la sécurité au niveau des lignes est activée si les valeurs de l'une des options de configuration suivantes ne correspondent pas à la valeur par défaut de la session :
  - `enable_case_sensitive_super_attribute`
  - `enable_case_sensitive_identifier`
  - `downcase_delimited_identifier`

  Envisagez de réinitialiser les options de configuration de votre session si vous tentez d'interroger une relation avec sécurité au niveau des lignes et que vous voyez le message « La relation protégée RLS ne prend pas en charge la configuration au niveau de la session si la sensibilité à la casse est différente de sa valeur par défaut ».

- Lorsque votre cluster ou votre espace de noms sans serveur mis en service est soumis à des politiques de sécurité au niveau des lignes, les commandes suivantes sont bloquées pour les utilisateurs standard :

  ```
  ALTER <current_user> SET enable_case_sensitive_super_attribute/
  enable_case_sensitive_identifier/downcase_delimited_identifier
  ```

  Lorsque vous créez des politiques RLS, nous vous recommandons de modifier les paramètres des options de configuration par défaut pour les utilisateurs standard afin qu'ils correspondent aux paramètres des options de configuration de la session au moment où la politique a été créée. Les super-utilisateurs et les utilisateurs dotés du privilège ALTER USER peuvent faire cela en utilisant les paramètres de groupe de paramètres ou la commande ALTER USER. Pour en savoir plus sur les groupes de paramètres, consultez Groupes de paramètres Amazon Redshift dans le Guide de gestion Amazon Redshift. Pour en savoir plus sur la commande ALTER USER, consultez ALTER USER.
- Les vues et les vues à liaison tardive (LBV) disposant de politiques de sécurité au niveau des lignes ne peuvent pas être remplacées par des utilisateurs ordinaires utilisant la commande CREATE VIEW. Pour remplacer des vues ou par LBVs des politiques RLS, détachez d'abord toutes les politiques RLS qui leur sont associées, remplacez les vues ou LBVs attachez à nouveau les politiques. Les superutilisateurs et les utilisateurs dotés du `sys:secadmin permission` peuvent utiliser CREATE VIEW sur des vues ou LBVs avec des politiques RLS sans détacher les politiques.
- Les vues disposant de politiques de sécurité au niveau des lignes ne peuvent pas référencer les tables système ni les vues système.
- Une vue à liaison tardive référencée par une vue normale ne peut pas être protégée par RLS.
- Les relations protégées par RLS et les données imbriquées provenant de lacs de données ne sont pas accessibles dans la même requête.

Bonnes pratiques pour la performance de la sécurité RLS

Voici les bonnes pratiques pour garantir de meilleures performances d'Amazon Redshift sur les tables protégées par RLS.

Sécurité des opérateurs et des fonctions

Lors de l'interrogation de tables protégées par RLS, l'utilisation de certains opérateurs ou de certaines fonctions peut entraîner une dégradation des performances. Amazon Redshift classe les opérateurs et les fonctions comme sécurisés ou non sécurisés pour interroger les tables protégées par RLS. Une fonction ou un opérateur est classé comme étant sécurisé par RLS lorsqu'il ne présente aucun effet secondaire observable en fonction des entrées. En particulier, une fonction ou un opérateur sécurisé par RLS ne peut pas correspondre à l'un des cas suivants :

- Fournit une valeur d'entrée, ou toute valeur qui dépend de la valeur d'entrée, avec ou sans message d'erreur.
- Echoue ou renvoie des erreurs qui dépendent de la valeur d'entrée.

Les opérateurs non sécurisés par RLS comprennent :

- Les opérateurs arithmétiques — +, -, /, *, %.
- Les opérateurs de texte – LIKE et SIMILAR TO.
- Les opérateurs de distribution.
- UDFs.

Utilisez l'instruction SELECT suivante pour vérifier la sécurité des opérateurs et des fonctions.

```
SELECT proname, proc_is_rls_safe(oid) FROM pg_proc;
```

Amazon Redshift impose des restrictions sur l'ordre d'évaluation des prédicats de l'utilisateur contenant des opérateurs et des fonctions non sécurisés par RLS lors de la planification de requêtes sur des tables protégées par RLS. Les requêtes faisant référence à des opérateurs ou des fonctions non sécurisés par RLS peuvent entraîner une dégradation des performances lors de l'interrogation de tables protégées par RLS. Les performances peuvent se dégrader de manière significative lorsqu'Amazon Redshift ne parvient pas à déléguer les prédicats non sécurisés par RLS aux analyses de la table de base pour tirer parti des clés de tri. Pour de meilleures performances, évitez les requêtes utilisant des prédicats non sécurisés par RLS qui tirent parti d'une clé de tri. Pour vérifier qu'Amazon Redshift est capable de déléguer des opérateurs et des fonctions, vous pouvez utiliser des instructions EXPLAIN en association avec l'autorisation système EXPLAIN RLS.

Mise en cache du résultat

Pour raccourcir le temps d'exécution des requêtes et améliorer les performances du système, Amazon Redshift met en cache les résultats de certains types de requêtes dans la mémoire du nœud principal.

Amazon Redshift utilise les résultats mis en cache pour une nouvelle requête analysant les tables protégées par RLS si l'ensemble des conditions pour les tables non protégées et les éléments suivants sont satisfaits :

- Les tables ou les vues incluses dans la politique n'ont pas été modifiées.
- La politique n'utilise pas une fonction qui nécessite une évaluation à chaque exécution telle que GETDATE ou CURRENT_USER.

Pour de meilleures performances, évitez d'utiliser des prédicats de politique qui ne satisfont pas les conditions ci-dessus.

Pour plus d'informations sur la mise en cache des résultats dans Amazon Redshift, consultez Mise en cache du résultat .

politiques complexes

Pour de meilleures performances, évitez d'utiliser des politiques complexes avec des sous-requêtes qui rejoignent plusieurs tables.

Exemple de sécurité au niveau des lignes end-to-end

L' end-to-endexemple suivant illustre la façon dont un superutilisateur crée des utilisateurs et des rôles. Ensuite, un utilisateur disposant du rôle secadmin crée, attache, détache et supprime les politiques RLS. Cet exemple utilise l'exemple de base données tickit. Pour plus d'informations, consultez Charger les données d'Amazon S3 vers Amazon Redshift dans le Guide de démarrage d'Amazon Redshift.

```
-- Create users and roles referenced in the policy statements.
CREATE ROLE analyst;
CREATE ROLE consumer;
CREATE ROLE dbadmin;
CREATE ROLE auditor;
CREATE USER bob WITH PASSWORD 'Name_is_bob_1';
CREATE USER alice WITH PASSWORD 'Name_is_alice_1';
```

```
CREATE USER joe WITH PASSWORD 'Name_is_joe_1';
CREATE USER molly WITH PASSWORD 'Name_is_molly_1';
CREATE USER bruce WITH PASSWORD 'Name_is_bruce_1';
GRANT ROLE sys:secadmin TO bob;
GRANT ROLE analyst TO alice;
GRANT ROLE consumer TO joe;
GRANT ROLE dbadmin TO molly;
GRANT ROLE auditor TO bruce;
GRANT ALL ON TABLE tickit_category_redshift TO PUBLIC;
GRANT ALL ON TABLE tickit_sales_redshift TO PUBLIC;
GRANT ALL ON TABLE tickit_event_redshift TO PUBLIC;

-- Create table and schema referenced in the policy statements.
CREATE SCHEMA target_schema;
GRANT ALL ON SCHEMA target_schema TO PUBLIC;
CREATE TABLE target_schema.target_event_table (LIKE tickit_event_redshift);
GRANT ALL ON TABLE target_schema.target_event_table TO PUBLIC;

-- Change session to analyst alice.
SET SESSION AUTHORIZATION alice;

-- Check the tuples visible to analyst alice.
-- Should contain all 3 categories.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Concerts');

SELECT poldb, polname, polalias, polatts, polqual, polenabled, polmodifiedby FROM
 svv_rls_policy WHERE poldb = CURRENT_DATABASE();

ATTACH RLS POLICY policy_concerts ON tickit_category_redshift TO ROLE analyst, ROLE
 dbadmin;

ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;

SELECT * FROM svv_rls_attached_policy;
```

```
-- Change session to analyst alice.
SET SESSION AUTHORIZATION alice;

-- Check that tuples with only `Concert` category will be visible to analyst alice.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to consumer joe.
SET SESSION AUTHORIZATION joe;

-- Although the policy is attached to a different role, no tuples will be
-- visible to consumer joe because the default deny all policy is applied.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to dbadmin molly.
SET SESSION AUTHORIZATION molly;

-- Check that tuples with only `Concert` category will be visible to dbadmin molly.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Check that EXPLAIN output contains RLS SecureScan to prevent disclosure of
-- sensitive information such as RLS filters.
EXPLAIN SELECT catgroup, count(*) FROM tickit_category_redshift GROUP BY catgroup ORDER
 BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

-- Grant IGNORE RLS permission so that RLS policies do not get applicable to role
 dbadmin.
GRANT IGNORE RLS TO ROLE dbadmin;

-- Grant EXPLAIN RLS permission so that anyone in role auditor can view complete
 EXPLAIN output.
GRANT EXPLAIN RLS TO ROLE auditor;

-- Change session to dbadmin molly.
SET SESSION AUTHORIZATION molly;
```

```
-- Check that all tuples are visible to dbadmin molly because `IGNORE RLS` is granted
 to role dbadmin.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to auditor bruce.
SET SESSION AUTHORIZATION bruce;

-- Check explain plan is visible to auditor bruce because `EXPLAIN RLS` is granted to
 role auditor.
EXPLAIN SELECT catgroup, count(*) FROM tickit_category_redshift GROUP BY catgroup ORDER
 BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

DETACH RLS POLICY policy_concerts ON tickit_category_redshift FROM ROLE analyst, ROLE
 dbadmin;

-- Change session to analyst alice.
SET SESSION AUTHORIZATION alice;

-- Check that no tuples are visible to analyst alice.
-- Although the policy is detached, no tuples will be visible to analyst alice
-- because of default deny all policy is applied if the table has RLS on.
SELECT catgroup, count(*)
FROM tickit_category_redshift
GROUP BY catgroup ORDER BY catgroup;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

CREATE RLS POLICY policy_events
WITH (eventid INTEGER) AS ev
USING (
    ev.eventid IN (SELECT eventid FROM tickit_sales_redshift WHERE qtysold <3)
);

ATTACH RLS POLICY policy_events ON tickit_event_redshift TO ROLE analyst;
ATTACH RLS POLICY policy_events ON target_schema.target_event_table TO ROLE consumer;

RESET SESSION AUTHORIZATION;
```

```
-- Can not cannot alter type of dependent column.
ALTER TABLE target_schema.target_event_table ALTER COLUMN eventid TYPE float;
ALTER TABLE tickit_event_redshift ALTER COLUMN eventid TYPE float;
ALTER TABLE tickit_sales_redshift ALTER COLUMN eventid TYPE float;
ALTER TABLE tickit_sales_redshift ALTER COLUMN qtysold TYPE float;

-- Can not cannot rename dependent column.
ALTER TABLE target_schema.target_event_table RENAME COLUMN eventid TO renamed_eventid;
ALTER TABLE tickit_event_redshift RENAME COLUMN eventid TO renamed_eventid;
ALTER TABLE tickit_sales_redshift RENAME COLUMN eventid TO renamed_eventid;
ALTER TABLE tickit_sales_redshift RENAME COLUMN qtysold TO renamed_qtysold;

-- Can not drop dependent column.
ALTER TABLE target_schema.target_event_table DROP COLUMN eventid CASCADE;
ALTER TABLE tickit_event_redshift DROP COLUMN eventid CASCADE;
ALTER TABLE tickit_sales_redshift DROP COLUMN eventid CASCADE;
ALTER TABLE tickit_sales_redshift DROP COLUMN qtysold CASCADE;

-- Can not drop lookup table.
DROP TABLE tickit_sales_redshift CASCADE;

-- Change session to security administrator bob.
SET SESSION AUTHORIZATION bob;

DROP RLS POLICY policy_concerts;
DROP RLS POLICY IF EXISTS policy_events;

ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY OFF;

RESET SESSION AUTHORIZATION;

-- Drop users and roles.
DROP USER bob;
DROP USER alice;
DROP USER joe;
DROP USER molly;
DROP USER bruce;
DROP ROLE analyst;
DROP ROLE consumer;
DROP ROLE auditor FORCE;
DROP ROLE dbadmin FORCE;
```

Sécurité des métadonnées

À l'instar de la sécurité au niveau des lignes d'Amazon Redshift, la sécurité des métadonnées vous permet de contrôler plus précisément vos métadonnées. Si la sécurité des métadonnées est activée pour votre cluster provisionné ou votre groupe de travail sans serveur, les utilisateurs peuvent consulter les métadonnées des objets auxquels ils ont accès. La sécurité des métadonnées permet de séparer la visibilité en fonction des besoins. Par exemple, vous pouvez utiliser un seul entrepôt des données pour centraliser l'ensemble de votre stockage de données. Toutefois, si vous stockez des données pour plusieurs secteurs, la gestion de la sécurité peut s'avérer difficile. Lorsque la sécurité des métadonnées est activée, vous pouvez configurer la visibilité. Les utilisateurs d'un secteur peuvent avoir une meilleure visibilité sur leurs objets, tandis que vous limitez l'accès à l'affichage pour les utilisateurs d'un autre secteur. La sécurité des métadonnées prend en charge tous les types d'objets, comme les schémas, les tables, les vues, les vues matérialisées, les procédures stockées, les fonctions définies par l'utilisateur et les modèles de machine learning.

Les utilisateurs peuvent consulter les métadonnées des objets dans les circonstances suivantes :

- Si l'accès aux objets est accordé à l'utilisateur.
- Si l'accès aux objets est accordé à un groupe ou à un rôle dont l'utilisateur fait partie.
- Si l'objet est public.
- Si l'utilisateur est le propriétaire de l'objet de base de données.

Pour activer la sécurité des métadonnées, utilisez la commande ALTER SYSTEM. La syntaxe suivante montre comment utiliser la commande ALTER SYSTEM avec la sécurité des métadonnées.

```
ALTER SYSTEM SET metadata_security=[true|t|on|false|f|off];
```

Lorsque vous activez la sécurité des métadonnées, tous les utilisateurs disposant des autorisations nécessaires peuvent voir les métadonnées pertinentes des objets auxquels ils ont accès. Si vous souhaitez que seuls certains utilisateurs puissent accéder à la sécurité des métadonnées, accordez l'autorisation `ACCESS CATALOG` à un rôle, puis attribuez le rôle à l'utilisateur. Pour plus d'informations sur l'utilisation des rôles pour mieux contrôler la sécurité, consultez Contrôle d'accès basé sur les rôles.

L'exemple suivant montre comment accorder l'autorisation `ACCESS CATALOG` à un rôle, puis comment attribuer le rôle à un utilisateur. Pour plus d'informations sur l'octroi d'autorisations, consultez la commande GRANT.

```
CREATE ROLE sample_metadata_viewer;

GRANT ACCESS CATALOG TO ROLE sample_metadata_viewer;

GRANT ROLE sample_metadata_viewer to salesadmin;
```

Si vous préférez utiliser des rôles déjà définis, les rôles définis par le système `operator`, `secadmin`, `dba` et `superuser` disposent tous des autorisations nécessaires pour afficher les métadonnées des objets. Par défaut, les super-utilisateurs peuvent consulter le catalogue complet.

```
GRANT ROLE operator to sample_user;
```

Si vous utilisez des rôles pour contrôler la sécurité des métadonnées, vous avez accès à toutes les vues et fonctions du système associées au contrôle d'accès basé sur les rôles. Par exemple, vous pouvez interroger la vue SVV_ROLES pour voir tous les rôles. Pour savoir si un utilisateur est membre d'un rôle ou d'un groupe, utilisez la fonction USER_IS_MEMBER_OF. Pour une liste complète des vues SVV, consultez Vues de métadonnées SVV. Pour obtenir la liste des fonctions relatives aux informations système, consultez Fonctions d'informations système.

Masquage dynamique des données

Grâce au masquage des données (DDM) dans Amazon Redshift, vous pouvez protéger les données sensibles dans votre entrepôt des données. Vous pouvez modifier la façon dont Amazon Redshift montre les données sensibles à l'utilisateur au moment de la requête, sans les transformer dans la base de données. Vous contrôlez l'accès aux données par le biais de politiques de masquage qui appliquent des règles de masquage personnalisées à un utilisateur ou à un rôle donné. Vous pouvez ainsi répondre à l'évolution des exigences de confidentialité sans modifier les données sous-jacentes ni modifier les requêtes SQL.

Les politiques de masquage dynamique des données cachent, masquent ou pseudonymisent les données qui correspondent à un format donné. Lorsqu'elle est attachée à une table, l'expression de masquage est appliquée à une ou plusieurs de ses colonnes. Vous pouvez modifier davantage les politiques de masquage pour ne les appliquer qu'à certains utilisateurs ou à des rôles définis par l'utilisateur que vous pouvez créer avec Contrôle d'accès basé sur les rôles (RBAC). En outre, vous pouvez appliquer le DDM au niveau de la cellule en utilisant des colonnes conditionnelles lors de la création de votre politique de masquage. Pour plus d'informations sur le masquage conditionnel, consultez Masquage conditionnel des données dynamiques.

Vous pouvez appliquer plusieurs politiques de masquage avec différents niveaux de masquage à la même colonne d'une table et les attribuer à différents rôles. Pour éviter les conflits lorsque vous avez des rôles différents et que des politiques différentes s'appliquent à une colonne, vous pouvez définir des priorités pour chaque application. Vous pouvez ainsi contrôler les données auxquelles un utilisateur ou un rôle donné peut accéder. Les politiques DDM peuvent supprimer partiellement ou complètement des données, ou les hacher à l'aide de fonctions définies par l'utilisateur écrites en SQL, Python ou avec AWS Lambda. En masquant les données à l'aide de hachages, vous pouvez appliquer des jointures à ces données sans accéder à des informations potentiellement sensibles.

Commandes SQL pour gérer les politiques de masquage dynamique des données

Vous pouvez effectuer les actions suivantes pour créer, joindre, détacher et supprimer des politiques de masquage dynamique des données :

- Pour créer une politique DDM, utilisez la commande CREATE MASKING POLICY.

  Voici un exemple de création d'une politique de masquage à l'aide d'une fonction de hachage SHA-2.

```
CREATE MASKING POLICY hash_credit
WITH (credit_card varchar(256))
USING (sha2(credit_card + 'testSalt', 256));
```

- Pour modifier une politique DDM existante, utilisez la commande MODIFIER LA POLITIQUE DE MASQUAGE.

  Voici un exemple de modification d'une politique de masquage existante.

```
ALTER MASKING POLICY hash_credit
USING (sha2(credit_card + 'otherTestSalt', 256));
```

- Pour attacher une politique DDM sur une table à un ou plusieurs utilisateurs ou rôles, utilisez la commande ATTACH MASKING POLICY.

  Voici un exemple d'attachement d'une politique de masquage à une paire colonne/rôle.

```
 ATTACH MASKING POLICY hash_credit
ON credit_cards (credit_card)
TO ROLE science_role
```

```
PRIORITY 30;
```

La clause PRIORITY détermine quelle politique de masquage s'applique à une session utilisateur lorsque plusieurs politiques sont attachées à la même colonne. Par exemple, si l'utilisateur de l'exemple précédent a une autre politique de masquage attachée à la même colonne de carte de crédit avec une priorité de 20, la politique de science_role est celle qui s'applique, car elle possède la priorité la plus élevée (30).

- Pour détacher une politique DDM sur une table d'un ou de plusieurs utilisateurs ou rôles, utilisez la commande DETACH MASKING POLICY.

  Voici un exemple de détachement d'une politique de masquage à partir d'une paire colonne/rôle.

```
DETACH MASKING POLICY hash_credit
ON credit_cards(credit_card)
FROM ROLE science_role;
```

- Pour supprimer une politique DDM de toutes les bases de données, utilisez la commande DROP MASKING POLICY.

  Voici un exemple de suppression d'une politique de masquage de toutes les bases de données.

```
DROP MASKING POLICY hash_credit;
```

Hiérarchie des politiques de masquage dynamique des données

Tenez compte des éléments suivants lorsque vous attachez plusieurs politiques de masquage :

- Vous pouvez attacher plusieurs politiques de masquage à une seule colonne.
- Lorsque plusieurs politiques de masquage sont applicables à une requête, la politique de priorité la plus élevée attachée à chaque colonne respective s'applique. Prenez l'exemple de code suivant.

```
ATTACH MASKING POLICY partial_hash
ON credit_cards(address, credit_card)
TO ROLE analytics_role
PRIORITY 20;

ATTACH MASKING POLICY full_hash
ON credit_cards(credit_card, ssn)
TO ROLE auditor_role
```

```
PRIORITY 30;

SELECT address, credit_card, ssn
FROM credit_cards;
```

Lors de l'exécution de l'instruction SELECT, un utilisateur ayant à la fois les rôles d'analyste et d'auditeur voit la colonne d'adresse dans laquelle la politique de masquage `partial_hash` appliquée. Il voit les colonnes relatives aux cartes de crédit et au SSN auxquelles la politique de masquage `full_hash` est appliquée, car la politique `full_hash` a la priorité la plus élevée sur la colonne des cartes de crédit.

- Si vous ne spécifiez pas de priorité quand vous attachez une politique de masquage, la priorité par défaut est de 0.
- Vous ne pouvez pas attacher deux politiques à la même colonne avec la même priorité.
- Vous ne pouvez pas attacher deux politiques à la même combinaison d'utilisateur et de colonne, ou de rôle et de colonne.
- Lorsque plusieurs politiques de masquage sont applicables le long du même chemin SUPER alors qu'elles sont attachées au même utilisateur ou rôle, seul l'attachement ayant la plus haute priorité prend effet. Considérez les exemples suivants.

  Le premier exemple montre deux politiques de masquage attachées au même chemin, celle ayant la priorité la plus élevée prenant effet.

  ```
  ATTACH MASKING POLICY hide_name
  ON employees(col_person.name)
  TO PUBLIC
  PRIORITY 20;

  ATTACH MASKING POLICY hide_last_name
  ON employees(col_person.name.last)
  TO PUBLIC
  PRIORITY 30;

  --Only the hide_last_name policy takes effect.
  SELECT employees.col_person.name FROM employees;
  ```

  Le deuxième exemple montre deux politiques de masquage attachées à différents chemins dans le même objet SUPER, sans conflit entre les politiques. Les deux attachements s'appliqueront en même temps.

```
ATTACH MASKING POLICY hide_first_name
ON employees(col_person.name.first)
TO PUBLIC
PRIORITY 20;

ATTACH MASKING POLICY hide_last_name
ON employees(col_person.name.last)
TO PUBLIC
PRIORITY 20;

--Both col_person.name.first and col_person.name.last are masked.
SELECT employees.col_person.name FROM employees;
```

Pour confirmer quelle politique de masquage s'applique à un utilisateur donné et à une colonne ou à une combinaison de rôles et de colonnes, les utilisateurs dotés du `sys:secadmin`rôle peuvent rechercher la column/role or column/user paire dans la vue SVV_ATTACHED_MASKING_POLICY système. Pour de plus amples informations, veuillez consulter Vues dynamiques du système de masquage des données.

Utilisation du masquage dynamique des données avec des chemins de type de données SUPER

Amazon Redshift permet d'attacher des politiques de masquage dynamique des données aux chemins des colonnes de type SUPER. Pour plus d'informations sur le type de données SUPER, consultez Données semi-structurées dans Amazon Redshift.

Lorsque vous associez des politiques de masquage aux chemins de colonnes de type SUPER, tenez compte des points suivants.

- Lorsque vous associez une politique de masquage à un chemin dans une colonne, cette colonne doit être définie comme type de données SUPER. Vous ne pouvez appliquer des politiques de masquage qu'aux valeurs scalaires du chemin SUPER. Vous ne pouvez pas appliquer de politiques de masquage à des structures ou à des tableaux complexes.
- Vous pouvez appliquer différentes politiques de masquage à plusieurs valeurs scalaires sur une seule colonne SUPER, à condition que les chemins SUPER ne soient pas en conflit. Par exemple, les chemins SUPER `a.b` et `a.b.c` entrent en conflit parce qu'ils se trouvent sur le même chemin, `a.b` étant le parent de `a.b.c`. Les chemins SUPER `a.b.c` et `a.b.d` ne sont pas en conflit.

- Amazon Redshift ne peut pas vérifier que les chemins auxquels s'attache une politique de masquage existent dans les données et sont du type attendu, tant que la politique n'est pas appliquée à l'exécution de la requête utilisateur. Par exemple, lorsque vous attachez une politique de masquage qui masque les valeurs TEXT à un chemin SUPER contenant une valeur INT, Amazon Redshift essaie de convertir le type de valeur sur le chemin.

  Dans de telles situations, le comportement d'Amazon Redshift au moment de l'exécution dépend de vos paramètres de configuration pour interroger les objets SUPER. Par défaut, Amazon Redshift est en mode laxiste et résoudra les chemins manquants et les conversions non valides comme `NULL` pour le chemin SUPER indiqué. Pour plus d'informations sur les paramètres de configuration concernant SUPER, consultez Configurations SUPER.
- SUPER est un type sans schéma, ce qui signifie qu'Amazon Redshift ne peut pas confirmer l'existence de la valeur sur un chemin SUPER donné. Si vous attachez une politique de masquage à un chemin SUPER qui n'existe pas et qu'Amazon Redshift est en mode laxiste, Amazon Redshift résoudra le chemin à une valeur `NULL`. Nous vous recommandons de tenir compte du format attendu des objets SUPER et de la probabilité qu'ils présentent des attributs inattendus lorsque vous attachez des politiques de masquage aux chemins des colonnes SUPER. Si vous pensez qu'il existe un schéma inattendu dans votre colonne SUPER, pensez à attacher vos politiques de masquage directement à la colonne SUPER. Vous pouvez utiliser les fonctions d'information de type SUPER pour vérifier les attributs et les types, et utiliser `OBJECT_TRANSFORM` pour masquer les valeurs. Pour plus d'informations sur les fonctions d'information de type SUPER, consultez Fonctions d'informations sur le type SUPER.

Exemples

Association de politiques de masquage aux chemins SUPER

L'exemple suivant attache plusieurs politiques de masquage à plusieurs chemins de type SUPER dans une colonne.

```
CREATE TABLE employees (
    col_person SUPER
);

INSERT INTO employees
VALUES
    (
        json_parse('
            {
```

```
                "name": {
                    "first": "John",
                    "last": "Doe"
                },
                "age": 25,
                "ssn": "111-22-3333",
                "company": "Company Inc."
            }
        ')
    ),
    (
        json_parse('
            {
                "name": {
                    "first": "Jane",
                    "last": "Appleseed"
                },
                "age": 34,
                "ssn": "444-55-7777",
                "company": "Organization Org."
            }
        ')
    )
;
GRANT ALL ON ALL TABLES IN SCHEMA "public" TO PUBLIC;

-- Create the masking policies.

-- This policy converts the given name to all uppercase letters.
CREATE MASKING POLICY mask_first_name
WITH(first_name TEXT)
USING ( UPPER(first_name) );

-- This policy replaces the given name with the fixed string 'XXXX'.
CREATE MASKING POLICY mask_last_name
WITH(last_name TEXT)
USING ( 'XXXX'::TEXT );

-- This policy rounds down the given age to the nearest 10.
CREATE MASKING POLICY mask_age
WITH(age INT)
USING ( (FLOOR(age::FLOAT / 10) * 10)::INT );

-- This policy converts the first five digits of the given SSN to 'XXX-XX'.
```

```
CREATE MASKING POLICY mask_ssn
WITH(ssn TEXT)
USING ( 'XXX-XX-'::TEXT || SUBSTRING(ssn::TEXT FROM 8 FOR 4) );

-- Attach the masking policies to the employees table.
ATTACH MASKING POLICY mask_first_name
ON employees(col_person.name.first)
TO PUBLIC;

ATTACH MASKING POLICY mask_last_name
ON employees(col_person.name.last)
TO PUBLIC;

ATTACH MASKING POLICY mask_age
ON employees(col_person.age)
TO PUBLIC;

ATTACH MASKING POLICY mask_ssn
ON employees(col_person.ssn)
TO PUBLIC;

-- Verify that your masking policies are attached.
SELECT
    policy_name,
    TABLE_NAME,
    priority,
    input_columns,
    output_columns
FROM
    svv_attached_masking_policy;

   policy_name   | table_name | priority |           input_columns           |
 output_columns
-----------------+------------+----------+-----------------------------------
+-----------------------------------
 mask_age        | employees  |        0 | ["col_person.\"age\""]            |
 ["col_person.\"age\""]
 mask_first_name | employees  |        0 | ["col_person.\"name\".\"first\""] |
 ["col_person.\"name\".\"first\""]
 mask_last_name  | employees  |        0 | ["col_person.\"name\".\"last\""]  |
 ["col_person.\"name\".\"last\""]
 mask_ssn        | employees  |        0 | ["col_person.\"ssn\""]            |
 ["col_person.\"ssn\""]
(4 rows)
```

```
-- Observe the masking policies taking effect.
SELECT col_person FROM employees ORDER BY col_person.age;

-- This result is formatted for ease of reading.
         col_person
-------------------------------
{
    "name": {
        "first": "JOHN",
        "last": "XXXX"
    },
    "age": 20,
    "ssn": "XXX-XX-3333",
    "company": "Company Inc."
}
{
    "name": {
        "first": "JANE",
        "last": "XXXX"
    },
    "age": 30,
    "ssn": "XXX-XX-7777",
    "company": "Organization Org."
}
```

Voici quelques exemples d’attachements non valides de politique de masquage à des chemins SUPER.

```
-- This attachment fails because there is already a policy
-- with equal priority attached to employees.name.last, which is
-- on the same SUPER path as employees.name.
ATTACH MASKING POLICY mask_ssn
ON employees(col_person.name)
TO PUBLIC;
ERROR:  DDM policy "mask_last_name" is already attached on relation "employees" column
 "col_person."name"."last"" with same priority

-- Create a masking policy that masks DATETIME objects.
CREATE MASKING POLICY mask_date
WITH(INPUT DATETIME)
USING ( INPUT );
```

```
-- This attachment fails because SUPER type columns can't contain DATETIME objects.
ATTACH MASKING POLICY mask_date
ON employees(col_person.company)
TO PUBLIC;
ERROR:  cannot attach masking policy for output of type "timestamp without time zone"
 to column "col_person."company"" of type "super
```

Voici un exemple d'attachement d'une politique de masquage à un chemin SUPER qui n'existe pas. Par défaut, Amazon Redshift résoudra le chemin à la valeur NULL.

```
ATTACH MASKING POLICY mask_first_name
ON employees(col_person.not_exists)
TO PUBLIC;

SELECT col_person FROM employees LIMIT 1;

-- This result is formatted for ease of reading.
         col_person
-----------------------------------
{
    "name": {
        "first": "JOHN",
        "last": "XXXX"
    },
    "age": 20,
    "ssn": "XXX-XX-3333",
    "company": "Company Inc.",
    "not_exists": null
}
```

Masquage conditionnel des données dynamiques

Vous pouvez masquer les données au niveau de la cellule en créant des politiques de masquage avec des expressions conditionnelles dans l'expression de masquage. Par exemple, vous pouvez créer une politique de masquage qui applique différents masques à une valeur, en fonction de la valeur d'une autre colonne de cette ligne.

Voici un exemple d'utilisation du masquage conditionnel des données pour créer et attacher une politique de masquage qui supprime partiellement les numéros de carte de crédit impliqués dans des fraudes, tout en masquant complètement tous les autres numéros de carte de crédit. Vous devez être un super-utilisateur ou avoir le rôle `sys:secadmin` requis pour exécuter cet exemple.

```
--Create an analyst role.
CREATE ROLE analyst;

--Create a credit card table. The table contains an is_fraud boolean column,
--which is TRUE if the credit card number in that row was involved in a fraudulent
 transaction.
CREATE TABLE credit_cards (id INT, is_fraud BOOLEAN, credit_card_number VARCHAR(16));

--Create a function that partially redacts credit card numbers.
CREATE FUNCTION REDACT_CREDIT_CARD (credit_card VARCHAR(16))
RETURNS VARCHAR(16) IMMUTABLE
AS $$
    import re
    regexp = re.compile("^([0-9]{6})[0-9]{5,6}([0-9]{4})")

    match = regexp.search(credit_card)
    if match != None:
        first = match.group(1)
        last = match.group(2)
    else:
        first = "000000"
        last = "0000"

    return "{}XXXXX{}".format(first, last)
$$ LANGUAGE plpythonu;

--Create a masking policy that partially redacts credit card numbers if the is_fraud
 value for that row is TRUE,
--and otherwise blanks out the credit card number completely.
CREATE MASKING POLICY card_number_conditional_mask
    WITH (fraudulent BOOLEAN, pan varchar(16))
    USING (CASE WHEN fraudulent THEN REDACT_CREDIT_CARD(pan)
                ELSE Null
           END);

--Attach the masking policy to the credit_cards/analyst table/role pair.
ATTACH MASKING POLICY card_number_conditional_mask ON credit_cards (credit_card_number)
 USING (is_fraud, credit_card_number)
 TO ROLE analyst PRIORITY 100;
```

Vues dynamiques du système de masquage des données

Les superutilisateurs, les utilisateurs dotés du `sys:operator` rôle et les utilisateurs dotés de l'autorisation ACCESS SYSTEM TABLE peuvent accéder aux vues système relatives au DDM suivantes.

- SVV_MASKING_POLICY

  Utilisez SVV_MASKING_POLICY pour afficher toutes les politiques de masquage créées sur le cluster ou le groupe de travail.
- SVV_ATTACHED_MASKING_POLICY

  Utilisez SVV_ATTACHED_MASKING_POLICY pour afficher toutes les relations et tous les utilisateurs ou rôles associés à des politiques sur la base de données actuellement connectée.
- SYS_APPLIED_MASKING_POLICY_LOG

  Utilisez SYS_APPLIED_MASKING_POLICY_LOG pour suivre l'application des politiques de masquage aux requêtes qui font référence à des relations protégées par DDM.

Voici quelques exemples d’informations que vous pouvez trouver à l’aide des vues système.

```
--Select all policies associated with specific users, as opposed to roles
SELECT policy_name,
       schema_name,
       table_name,
       grantee
FROM svv_attached_masking_policy
WHERE grantee_type = 'user';

--Select all policies attached to a specific user
SELECT policy_name,
       schema_name,
       table_name,
       grantee
FROM svv_attached_masking_policy
WHERE grantee = 'target_grantee_name'

--Select all policies attached to a given table
SELECT policy_name,
       schema_name,
       table_name,
```

```
       grantee
FROM svv_attached_masking_policy
WHERE table_name = 'target_table_name'
      AND schema_name = 'target_schema_name';

--Select the highest priority policy attachment for a given role
SELECT samp.policy_name,
       samp.priority,
       samp.grantee,
       smp.policy_expression
FROM svv_masking_policy AS smp
JOIN svv_attached_masking_policy AS samp
    ON samp.policy_name = smp.policy_name
WHERE
    samp.grantee_type = 'role' AND
    samp.policy_name = mask_get_policy_for_role_on_column(
        'target_schema_name',
        'target_table_name',
        'target_column_name',
        'target_role_name')
ORDER BY samp.priority desc
LIMIT 1;

--See which policy a specific user will see on a specific column in a given relation
SELECT samp.policy_name,
       samp.priority,
       samp.grantee,
       smp.policy_expression
FROM svv_masking_policy AS smp
JOIN svv_attached_masking_policy AS samp
    ON samp.policy_name = smp.policy_name
WHERE
    samp.grantee_type = 'role' AND
    samp.policy_name = mask_get_policy_for_user_on_column(
        'target_schema_name',
        'target_table_name',
        'target_column_name',
        'target_user_name')
ORDER BY samp.priority desc;

 --Select all policies attached to a given relation.
SELECT policy_name,
schema_name,
relation_name,
```

```
database_name
FROM sys_applied_masking_policy_log
WHERE relation_name = 'relation_name'
AND schema_name = 'schema_name';
```

Considérations relatives à l'utilisation du masquage dynamique des données

Considérez ce qui suit quand vous utilisez le masquage dynamique des données :

- Lorsqu'ils interrogent des objets créés à partir de tables, tels que des vues, les utilisateurs voient les résultats en fonction de leurs propres politiques de masquage, et non celles de l'utilisateur qui a créé les objets. Par exemple, un utilisateur ayant le rôle d'analyste interrogeant une vue créée par un secadmin obtiendrait des résultats avec des politiques de masquage associées au rôle d'analyste.
- Pour éviter que la commande EXPLAIN n'expose potentiellement des filtres de politique de masquage sensibles, seuls les utilisateurs disposant de l'autorisation SYS_EXPLAIN_DDM peuvent voir les politiques de masquage appliquées dans les sorties EXPLAIN. Les utilisateurs ne disposent pas de l'autorisation SYS_EXPLAIN_DDM par défaut.

  Voici la syntaxe permettant d'accorder l'autorisation à un rôle.

  ```
  GRANT EXPLAIN MASKING TO ROLE rolename
  ```

  Pour plus d'informations sur la commande EXPLAIN, consultez EXPLAIN.
- Les utilisateurs ayant des rôles différents peuvent voir des résultats différents en fonction des conditions de filtre ou des conditions de jointure utilisées. Par exemple, l'exécution d'une commande SELECT sur une table à l'aide d'une valeur de colonne spécifique échouera si l'utilisateur exécutant la commande applique une politique de masquage qui masque cette colonne.
- Les politiques DDM doivent être appliquées avant toute opération de prédicat ou toute projection. Les politiques de masquage peuvent inclure les éléments suivants :
  - Opérations constantes à faible coût, telles que la conversion d'une valeur en valeur nulle
  - Opérations à coût modéré telles que le hachage HMAC
  - Opérations coûteuses telles que les appels à des fonctions Lambda externes définies par l'utilisateur

À ce titre, nous vous recommandons d'utiliser des expressions de masquage simples lorsque cela est possible.

- Vous pouvez utiliser des politiques DDM pour des rôles dotés de politiques de sécurité au niveau des lignes, mais notez que les politiques RLS sont appliquées avant le DDM. Une expression de masquage dynamique des données ne pourra pas lire une ligne protégée par RLS. Pour plus d'informations sur RLS, consultez Sécurité au niveau des lignes.
- Lorsque vous utilisez la commande COPY pour copier des tables parquet vers des tables cible protégées, vous devez spécifier explicitement les colonnes dans l'instruction COPY. Pour plus d'informations sur le mappage de colonnes avec COPY, consultez Options de mappage de colonnes.
- Il n'est pas possible d'attacher les politiques de DDM aux relations suivantes :
  - Tables et catalogues du système
  - Tables externes
  - Tables d'unités de partage des données
  - Vues matérialisées
  - Relations entre bases de données
  - Tables temporaires
  - Requêtes corrélées
- Les politiques de DDM peuvent contenir des tables de recherche. Des tables de recherche peuvent être présentes dans la clause USING. Les types de relations suivants ne peuvent pas être utilisés comme tables de recherche :
  - Tables et catalogues du système
  - Tables externes
  - Tables d'unités de partage des données
  - Vues, vues matérialisées et vues à liaison tardive
  - Relations entre bases de données
  - Tables temporaires
  - Requêtes corrélées

  Vous trouverez ci-dessous un exemple d'association d'une politique de masquage à une table de recherche.

```
--Create a masking policy referencing a lookup table
```

```
CREATE MASKING POLICY lookup_mask_credit_card WITH (credit_card TEXT) USING (
  CASE
    WHEN
      credit_card IN (SELECT credit_card_lookup FROM credit_cards_lookup)
    THEN '000000XXXX0000'
    ELSE REDACT_CREDIT_CARD(credit_card)
    END
  );

--Provides access to the lookup table via a policy attached to a role
GRANT SELECT ON TABLE credit_cards_lookup TO MASKING POLICY lookup_mask_credit_card;
```

- Vous ne pouvez pas attacher de politique de masquage qui produirait une sortie incompatible avec le type et la taille de la colonne cible. Par exemple, vous ne pouvez pas attacher une politique de masquage qui génère une chaîne de 12 caractères dans une colonne VARCHAR(10). Amazon Redshift prend en charge les exceptions suivantes :
  - Une politique de masquage avec le type d'entrée INTN peut être attachée à une politique de taille INTM tant que M < N. Par exemple, une politique d'entrée BIGINT (INT8) peut être attachée à une colonne smallint (). INT4
  - Une politique de masquage avec le type d’entrée NUMERIC ou DECIMAL peut toujours être attachée à une colonne FLOAT.
- Les politiques DDM ne peuvent pas être utilisées avec le partage de données. Si le producteur de données de l’unité de partage des données associe une politique DDM à une table de l’unité de partage des données, la table devient inaccessible aux utilisateurs du consommateur de données qui tentent d’interroger la table. Les tables auxquelles sont associées des politiques DDM ne peuvent pas être ajoutées à une unité de partage des données.
- Amazon Redshift ne prend pas en charge le partage de données avec DDM. Si le DDM pour les partages de données est activé dans une relation, la tentative d'ajout de la relation à un partage de données échoue sur le cluster ou l'espace de noms côté producteur avec l'erreur suivante :

  ```
  <ddm_protected_relation> or a relation dependent on it is protected by a masking
   policy and cannot be added to a datashare
  ```

  Si vous associez une politique de masquage à une relation côté producteur et que la relation est déjà incluse dans un partage de données, la tentative d'interrogation de la relation côté consommateur échoue avec l'erreur suivante :

  ```
  cross-cluster query of the masked relation <ddm_protected_relation> is not supported.
  ```

Vous pouvez désactiver DDM pour les partages de données à l'aide de la commande ALTER TABLE avec le paramètre MASKING OFF FOR DATASHARES. Pour de plus amples informations, veuillez consulter ALTER TABLE.

- Vous ne pouvez pas interroger les relations auxquelles des politiques DDM sont attachées si la valeur de l'une de vos options de configuration suivantes ne correspond pas à la valeur par défaut de la session :
  - `enable_case_sensitive_super_attribute`
  - `enable_case_sensitive_identifier`
  - `downcase_delimited_identifier`

  Envisagez de réinitialiser les options de configuration de votre session si vous tentez d'interroger une relation à laquelle une politique DDM est attachée et que vous voyez le message « La relation protégée DDM ne prend pas en charge la configuration au niveau de la session si la sensibilité à la casse est différente de sa valeur par défaut ».
- Lorsque votre cluster ou votre espace de noms sans serveur mis en service est soumis à des politiques de masquage dynamique des données, les commandes suivantes sont bloquées pour les utilisateurs standard :

  ```
  ALTER <current_user> SET enable_case_sensitive_super_attribute/
  enable_case_sensitive_identifier/downcase_delimited_identifier
  ```

  Lorsque vous créez des politiques DDM, nous vous recommandons de modifier les paramètres des options de configuration par défaut pour les utilisateurs standard afin qu'ils correspondent aux paramètres des options de configuration de la session au moment où la politique a été créée. Les super-utilisateurs et les utilisateurs dotés du privilège ALTER USER peuvent faire cela en utilisant les paramètres de groupe de paramètres ou la commande ALTER USER. Pour en savoir plus sur les groupes de paramètres, consultez Groupes de paramètres Amazon Redshift dans le Guide de gestion Amazon Redshift. Pour en savoir plus sur la commande ALTER USER, consultez ALTER USER.
- Les vues et les vues à liaison tardive (LBV) associées à des politiques de DDM ne peuvent pas être remplacées par des utilisateurs ordinaires utilisant la commande CREATE VIEW. Pour remplacer des vues ou par LBVs des politiques DDM, détachez d'abord toutes les politiques DDM qui leur sont associées, remplacez les vues ou LBVs attachez à nouveau les politiques. Les superutilisateurs et les utilisateurs `sys:secadmin` autorisés peuvent utiliser CREATE VIEW sur des vues ou LBVs avec des politiques DDM sans détacher les politiques.

- Les vues attachées à des politiques de DDM ne peuvent pas référencer les tables et les vues système. Les vues à liaison tardive peuvent faire référence à des tables et à des vues système.
- Les vues à liaison tardive attachées à des politiques de DDM ne peuvent pas faire référence à des données imbriquées dans des lacs de données, comme des documents JSON.
- Une vue à liaison tardive ne peut pas être attachée à des politiques de DDM si elle est référencée par n'importe quelle vue.
- Les politiques de DDM attachées aux vues à liaison tardive sont jointes par nom de colonne. Au moment de la requête, Amazon Redshift valide que toutes les politiques de masquage attachées à la vue à liaison tardive ont été appliquées correctement, et que le type de colonne de sortie de la vue à liaison tardive correspond aux types des politiques de masquage attachées. Si la validation échoue, Amazon Redshift renvoie une erreur pour la requête.
- Vous pouvez utiliser des variables de contexte de session personnalisées lors de la création de politiques DDM. L'exemple suivant définit les variables de contexte de session pour une politique DDM.

```
-- Set a customized context variable.
SELECT set_config('app.city', 'XXXX', FALSE);

-- Create a MASKING policy using current_setting() to get the value of a customized
 context variable.
CREATE MASKING POLICY city_mask
WITH (city VARCHAR(30))
USING (current_setting('app.city')::VARCHAR(30));

-- Attach the policy on the target table to one or more roles.
ATTACH MASKING POLICY city_mask
ON tickit_users_redshift(city)
TO ROLE analyst, ROLE dbadmin;
```

Pour plus de détails sur la façon de définir et de récupérer des variables de contexte de session personnalisées SETSET_CONFIG, reportez-vous àMONTRER,CURRENT_SETTING, etRESET. Pour plus d'informations sur la modification de la configuration du serveur en général, rendez-vous surModification de la configuration du serveur.

> **Important**
>
> Lorsque vous utilisez des variables de contexte de session dans les politiques DDM, la stratégie de sécurité dépend de l'utilisateur ou du rôle qui invoque la stratégie. Veillez à

éviter les failles de sécurité lorsque vous utilisez des variables de contexte de session dans les politiques DDM.

Exemple de masquage end-to-end dynamique de données

L' end-to-endexemple suivant montre comment créer et associer des politiques de masquage à une colonne. Ces règles permettent aux utilisateurs d'accéder à une colonne et de voir différentes valeurs, en fonction du degré de masquage des politiques associées à leurs rôles. Vous devez être un super-utilisateur ou avoir le rôle `sys:secadmin` requis pour exécuter cet exemple.

Création d'une politique de masquage

D'abord, créez une table et remplissez-la avec les valeurs des cartes de crédit.

```
--create the table
CREATE TABLE credit_cards (
  customer_id INT,
  credit_card TEXT
);

--populate the table with sample values
INSERT INTO credit_cards
VALUES
  (100, '4532993817514842'),
  (100, '4716002041425888'),
  (102, '5243112427642649'),
  (102, '6011720771834675'),
  (102, '6011378662059710'),
  (103, '373611968625635')
;

--run GRANT to grant permission to use the SELECT statement on the table
GRANT SELECT ON credit_cards TO PUBLIC;

--create two users
CREATE USER regular_user WITH PASSWORD '1234Test!';

CREATE USER analytics_user WITH PASSWORD '1234Test!';

--create the analytics_role role and grant it to analytics_user
--regular_user does not have a role
```

```
CREATE ROLE analytics_role;

GRANT ROLE analytics_role TO analytics_user;
```

Créez ensuite une politique de masquage à appliquer au rôle d'analyse.

```
--create a masking policy that fully masks the credit card number
CREATE MASKING POLICY mask_credit_card_full
WITH (credit_card VARCHAR(256))
USING ('000000XXXX0000'::TEXT);

--create a user-defined function that partially obfuscates credit card data
CREATE FUNCTION REDACT_CREDIT_CARD (credit_card TEXT)
RETURNS TEXT IMMUTABLE
AS $$
    import re
    regexp = re.compile("^([0-9]{6})[0-9]{5,6}([0-9]{4})")

    match = regexp.search(credit_card)
    if match != None:
        first = match.group(1)
        last = match.group(2)
    else:
        first = "000000"
        last = "0000"

    return "{}XXXXX{}".format(first, last)
$$ LANGUAGE plpythonu;

--create a masking policy that applies the REDACT_CREDIT_CARD function
CREATE MASKING POLICY mask_credit_card_partial
WITH (credit_card VARCHAR(256))
USING (REDACT_CREDIT_CARD(credit_card));

--confirm the masking policies using the associated system views
SELECT * FROM svv_masking_policy;

SELECT * FROM svv_attached_masking_policy;
```

Attachement d'une politique de masquage

Attachez les politiques de masquage à la table des cartes de crédit.

```
--attach mask_credit_card_full to the credit card table as the default policy
--all users will see this masking policy unless a higher priority masking policy is
 attached to them or their role
ATTACH MASKING POLICY mask_credit_card_full
ON credit_cards(credit_card)
TO PUBLIC;

--attach mask_credit_card_partial to the analytics role
--users with the analytics role can see partial credit card information
ATTACH MASKING POLICY mask_credit_card_partial
ON credit_cards(credit_card)
TO ROLE analytics_role
PRIORITY 10;

--confirm the masking policies are applied to the table and role in the associated
 system view
SELECT * FROM svv_attached_masking_policy;

--confirm the full masking policy is in place for normal users by selecting from the
 credit card table as regular_user
SET SESSION AUTHORIZATION regular_user;

SELECT * FROM credit_cards;

--confirm the partial masking policy is in place for users with the analytics role by
 selecting from the credit card table as analytics_user
SET SESSION AUTHORIZATION analytics_user;

SELECT * FROM credit_cards;
```

Modification d’une politique de masquage

La section suivante montre comment modifier une politique de masquage dynamique des données.

```
--reset session authorization to the default
RESET SESSION AUTHORIZATION;

--alter the mask_credit_card_full policy
ALTER MASKING POLICY mask_credit_card_full
USING ('00000000000000'::TEXT);

--confirm the full masking policy is in place after altering the policy, and that
 results are altered from '000000XXXX0000' to '00000000000000'
```

```
SELECT * FROM credit_cards;
```

Détachement et suppression d’une politique de masquage

La section suivante explique comment détacher et supprimer les politiques de masquage en supprimant toutes les politiques de masquage dynamique des données de la table.

```
--reset session authorization to the default
RESET SESSION AUTHORIZATION;

--detach both masking policies from the credit_cards table
DETACH MASKING POLICY mask_credit_card_full
ON credit_cards(credit_card)
FROM PUBLIC;

DETACH MASKING POLICY mask_credit_card_partial
ON credit_cards(credit_card)
FROM ROLE analytics_role;

--drop both masking policies
DROP MASKING POLICY mask_credit_card_full;

DROP MASKING POLICY mask_credit_card_partial;
```

Autorisations étendues

Les autorisations étendues vous permettent d'accorder des autorisations à un utilisateur ou à un rôle sur tous les objets d'un type au sein d'une base de données ou d'un schéma. Les utilisateurs et les rôles dotés d'autorisations étendues disposent des autorisations spécifiées sur tous les objets actuels et futurs de la base de données ou du schéma.

Vous pouvez consulter l'étendue des autorisations délimitées au niveau de la base de données dans. SVV_DATABASE_PRIVILEGES Vous pouvez consulter l'étendue des autorisations délimitées au niveau du schéma dans. SVV_SCHEMA_PRIVILEGES

Pour plus d’informations sur l’application d’autorisations étendues, consultez GRANT et REVOKE.

Considérations relatives à l’utilisation d’autorisations étendues

Tenez compte des éléments suivants lorsque vous utilisez des autorisations étendues :

- Vous pouvez utiliser des autorisations étendues pour accorder ou révoquer des autorisations sur l'étendue d'une base de données ou d'un schéma à destination ou en provenance d'un utilisateur ou d'un rôle spécifique.
- Vous ne pouvez pas accorder d'autorisations étendues à des groupes d'utilisateurs.
- L’octroi ou la révocation d’autorisations étendues modifie les autorisations pour tous les objets actuels et futurs du champ d’application.
- Les autorisations étendues et les autorisations au niveau de l'objet fonctionnent indépendamment les unes des autres. Par exemple, un utilisateur conservera les autorisations sur une table dans les deux cas suivants.
  - L'utilisateur dispose de l'autorisation SELECT sur la table schema1.table1 et de l'autorisation SELECT étendue sur schema1. L'utilisateur fait alors révoquer SELECT pour toutes les tables du schéma schema1. L'utilisateur conserve SELECT sur schema1.table1.
  - L'utilisateur dispose de l'autorisation SELECT sur la table schema1.table1 et de l'autorisation SELECT étendue sur schema1. L'utilisateur voit alors SELECT révoqué pour schema1.table1. L'utilisateur conserve SELECT sur schema1.table1.
- Pour accorder ou révoquer des autorisations étendues, vous devez répondre à l’un des critères suivants :
  - Super-utilisateurs
  - Utilisateurs disposant de l’option d’octroi de cette autorisation Pour plus d'informations sur les options de subvention, consultez le paramètre WITH GRANT OPTION dansGRANT.
- Les autorisations étendues ne peuvent être accordées ou révoquées que pour des objets de la base de données connectée, ou à partir de bases de données importées depuis une unité de partage des données.
- Vous pouvez utiliser des autorisations étendues pour définir les autorisations par défaut sur une base de données créée à partir d'un partage de données. Un utilisateur d’unité de partage des données côté consommateur disposant d’autorisations étendues sur une base de données partagée obtiendra automatiquement ces autorisations pour tout nouvel objet ajouté à l’unité de partage des données côté producteur.
- Les producteurs peuvent accorder des autorisations étendues sur les objets d'un schéma à un partage de données. (aperçu)

Référence SQL

Avec Amazon Redshift, vous pouvez tirer parti du langage SQL pour interroger et analyser efficacement de grandes quantités de données stockées dans votre entrepôt de données. La référence SQL couvre la syntaxe et l'utilisation des commandes SQL, des types de données, des fonctions, des opérateurs, etc., vous permettant d'extraire des informations et de prendre des décisions basées sur les données. Vous pouvez consulter cette référence pour écrire des requêtes optimisées, créer et gérer des objets de base de données et effectuer des transformations de données complexes. La référence suivante fournit des informations complètes sur l'utilisation de SQL dans l'environnement Amazon Redshift pour répondre à vos exigences en matière d'analyse de données.

Rubriques

- Amazon Redshift SQL
- Utilisation de SQL
- Commandes SQL
- Référence sur les fonctions SQL
- Mots réservés

Amazon Redshift SQL

Rubriques

- Fonctions SQL prises en charge sur le nœud principal
- Amazon Redshift et PostgreSQL

Amazon Redshift repose sur le langage SQL standard, avec des fonctionnalités supplémentaires permettant de gérer de très grands ensembles de données et de prendre en charge des analyses et des rapports performants sur ces données.

> **Note**
>
> La taille maximale d’une instruction SQL Amazon Redshift est de 16 Mo.

Fonctions SQL prises en charge sur le nœud principal

Certaines requêtes Amazon Redshift sont distribuées et exécutées sur les nœuds de calcul, et d'autres requêtes s'exécutent exclusivement sur le nœud principal.

Le nœud principal répartit SQL sur les nœuds de calcul chaque fois qu'une requête fait référence à des tables créées par l'utilisateur ou à des tables système (tables avec le préfixe STL ou STV, et vues système avec le préfixe SVL ou SVV). Une requête qui ne fait référence qu'aux tables catalogue (tables avec le préfixe PG, comme PG_TABLE_DEF, résidant sur le nœud principal) ou qui ne fait référence à aucune table, s'exécute exclusivement sur le nœud principal.

Certaines fonctions SQL d'Amazon Redshift sont prises en charge uniquement sur le nœud principal et ne le sont pas sur les nœuds de calcul. Une requête qui utilise une fonction de nœud principal doit être exécutée exclusivement sur celui-ci, et non sur les nœuds de calcul, sinon elle renverra une erreur.

La documentation de chaque fonction qui doit s'exécuter exclusivement sur le nœud principal comprend une note indiquant que la fonction renverra une erreur si elle fait référence à des tables définies par l'utilisateur ou à des tables du système Amazon Redshift. Consultez Fonctions exécutées uniquement sur le nœud principal pour obtenir la liste des fonctions qui s'exécutent exclusivement sur le nœud principal.

Exemples

Les exemples suivants utilisent l'exemple de base de données TICKIT. Pour plus d'informations sur l'exemple de base de données, rendez-vous surExemple de base de données.

CURRENT_SCHEMA

La fonction CURRENT_SCHEMA est une fonction de nœud principal uniquement. Dans cet exemple, comme la requête ne fait pas référence à une table, la fonction s'exécute exclusivement sur le nœud principal.

```
select current_schema();

current_schema
---------------
public
```

Dans l'exemple suivant, comme la requête fait référence à une table catalogue système, la fonction s'exécute exclusivement sur le nœud principal.

```
select * from pg_table_def
where schemaname = current_schema() limit 1;

 schemaname | tablename | column |   type   | encoding | distkey | sortkey | notnull
------------+-----------+--------+----------+----------+---------+---------+---------
 public     | category  | catid  | smallint | none     | t       |       1 | t
```

Dans l'exemple suivant, la requête fait référence à une table système Amazon Redshift qui réside sur les nœuds de calcul, et renvoie donc une erreur.

```
select current_schema(), userid from users;

INFO:  Function "current_schema()" not supported.
ERROR:  Specified types or functions (one per INFO message) not supported on Amazon
 Redshift tables.
```

SUBSTR

SUBSTR est également une fonction réservée aux nœuds principaux. Dans l'exemple suivant, la requête s'exécute exclusivement sur le nœud principal, car elle ne fait pas référence à une table.

```
SELECT SUBSTR('amazon', 5);

+--------+
| substr |
+--------+
| on     |
+--------+
```

Dans l'exemple suivant, la requête fait référence à une table qui réside sur les nœuds de calcul. Cela entraîne une erreur.

```
SELECT SUBSTR(catdesc, 1) FROM category LIMIT 1;

ERROR: SUBSTR() function is not supported (Hint: use SUBSTRING instead)
```

Pour exécuter correctement la requête précédente, utilisez SUBSTRING.

```
SELECT SUBSTRING(catdesc, 1) FROM category LIMIT 1;
```

```
+---------------------------------+
|            substring            |
+---------------------------------+
| National Basketball Association |
+---------------------------------+
```

FACTORIEL ()

FACTORIAL () est une fonction réservée aux nœuds leaders. Dans l'exemple suivant, la requête s'exécute exclusivement sur le nœud principal, car elle ne fait pas référence à une table.

```
SELECT FACTORIAL(5);

 factorial
-------------
 120
```

Dans l'exemple suivant, la requête fait référence à une table qui réside sur les nœuds de calcul. Cela entraîne une erreur lors de l'exécution à l'aide de l'éditeur de requêtes v2.

```
create table t(a int);
insert into t values (5);
select factorial(a) from t;

ERROR: Specified types or functions (one per INFO message) not supported on Redshift
 tables.
Info: Function "factorial(bigint)" not supported.
```

Amazon Redshift et PostgreSQL

Rubriques

- JDBC et ODBC Amazon Redshift et PostgreSQL
- Fonctions mises en œuvre différemment
- Fonctions PostgreSQL non prises en charge
- Types de données PostgreSQL non pris en charge
- Fonctions PostgreSQL non prises en charge

Amazon Redshift est basé sur PostgreSQL. Amazon Redshift et PostgreSQL présentent un certain nombre de différences très importantes dont vous devez être conscient lorsque vous concevez et développez vos applications d'entrepôt des données.

Amazon Redshift est spécifiquement conçu pour les applications de traitement analytique en ligne (OLAP) et de Business Intelligence (BI), qui nécessitent des requêtes complexes sur de grands ensembles de données. Parce qu'il répond à des exigences très différentes, le schéma de stockage de données spécialisé et le moteur d'exécution de requêtes qu'Amazon Redshift utilise sont complètement différents de l'implémentation PostgreSQL. Par exemple, là où les applications de traitement des transactions en ligne (OLTP) stockent généralement les données dans des lignes, Amazon Redshift stocke les données dans des colonnes, à l'aide d'encodages de compression de données spécialisés pour une utilisation optimale de la mémoire et des I/O disque. Certaines fonctions PostgreSQL adaptées au traitement OLTP à plus petite échelle, telles que les index secondaires et les opérations efficaces de manipulation de données sur une seule ligne, ont été omises pour améliorer les performances.

Consultez Architecture d'Amazon Redshift pour une explication détaillée de l'architecture du système d'entrepôt des données Amazon Redshift.

PostgreSQL 9.x comprend certaines fonctionnalités qui ne sont pas prises en charge par Amazon Redshift. En outre, il existe des différences importantes entre Amazon Redshift SQL et PostgreSQL dont vous devez être conscient. Cette section souligne les différences entre Amazon Redshift et PostgreSQL et fournit des conseils pour développer un entrepôt des données qui tire pleinement parti de l'implémentation SQL d'Amazon Redshift.

JDBC et ODBC Amazon Redshift et PostgreSQL

Amazon Redshift étant basé sur PostgreSQL, nous avons précédemment recommandé d'utiliser les pilotes PostgreSQL version 8.4.703 et JDBC4 PSQLodBC version 9.x. Si vous utilisez actuellement ces pilotes, nous vous recommandons de passer aux nouveaux pilotes spécifiques à Amazon Redshift. Pour plus d'informations sur les pilotes et la configuration des connexions, consultez Pilotes JDBC et ODBC pour Amazon Redshift dans le Guide de gestion Amazon Redshift.

Pour éviter les out-of-memory erreurs côté client lors de la récupération de grands ensembles de données à l'aide de JDBC, vous pouvez permettre à votre client de récupérer des données par lots en définissant le paramètre de taille d'extraction JDBC. Pour de plus amples informations, veuillez consulter Définition du paramètre de taille d'extraction JDBC.

Amazon Redshift ne reconnaît pas le paramètre JDBC maxRows. Spécifiez à la place une clause LIMIT afin de limiter le jeu de résultats. Vous pouvez aussi utiliser une clause OFFSET pour accéder directement à un point de départ spécifique du jeu de résultats.

Fonctions mises en œuvre différemment

De nombreux éléments du langage SQL d'Amazon Redshift ont des caractéristiques de performance différentes et utilisent une syntaxe et une sémantique très différentes de l'implémentation PostgreSQL équivalente.

> **Important**
>
> Ne présumez pas que la sémantique des éléments que Amazon Redshift et PostgreSQL ont en commun est identique. Veillez à consulter la rubrique Commandes SQL du Guide du développeur Amazon Redshift pour comprendre les différences, souvent subtiles.

Un exemple en particulier est celui de la commande VACUUM, qui permet de nettoyer et de réorganiser les tables. VACUUM fonctionne différemment que dans la version PostgreSQL et utilise un autre jeu de paramètres. Consultez Exécution de l'opération VACUUM sur les tables pour plus d'informations sur l'utilisation de VACUUM dans Amazon Redshift.

Souvent, les outils et les fonctions de gestion et d'administration de base de données sont également différents. Par exemple, Amazon Redshift maintient un ensemble de tables et de vues système qui fournissent des informations sur le fonctionnement du système. Pour plus d'informatons, consultez Vues de surveillance SYS.

La liste suivante comprend quelques exemples de fonctionnalités SQL qui sont implémentées différemment dans Amazon Redshift.

- CREATE TABLE

  Amazon Redshift ne prend pas en charge les espaces de table, le partitionnement de table et l'héritage, ainsi que certaines contraintes. L'implémentation Amazon Redshift de CREATE TABLE vous permet de définir les algorithmes de tri et de distribution pour les tables afin d'optimiser le traitement parallèle.

  Amazon Redshift Spectrum prend en charge le partitionnement des tables à l'aide de la commande CREATE EXTERNAL TABLE.

- ALTER TABLE

  Seul un sous-ensemble des actions ALTER COLUMN est pris en charge.

  ADD COLUMN ne prend en charge que l'ajout d'une seule colonne dans chaque instruction ALTER TABLE.

- COPY

  La commande Amazon Redshift COPY est hautement spécialisée pour permettre le chargement de données à partir de compartiments Amazon S3 et de tables Amazon DynamoDB et pour faciliter la compression automatique. Consultez la section Chargement de données dans Amazon Redshift et la référence sur la commande COPY pour plus de détails.

- VACUUM

  Les paramètres de VACUUM sont entièrement différents. Par exemple, dans PostgreSQL, l'opération VACUUM par défaut récupère simplement de l'espace et le rend disponible en vue de sa réutilisation ; cependant, l'opération VACUUM par défaut dans Amazon Redshift correspond à VACUUM FULL, qui récupère de l'espace disque et retrie toutes les lignes.

- Les espaces de fin des valeurs VARCHAR sont ignorés lorsque les valeurs de chaîne sont comparées. Pour de plus amples informations, veuillez consulter Signification des blancs de fin.

Fonctions PostgreSQL non prises en charge

Ces fonctionnalités PostgreSQL ne sont pas prises en charge par Amazon Redshift.

> **Important**
>
> Ne présumez pas que la sémantique des éléments que Amazon Redshift et PostgreSQL ont en commun est identique. Veillez à consulter la rubrique Commandes SQL du Guide du développeur Amazon Redshift pour comprendre les différences, souvent subtiles.

- L'outil de requête psql n'est pas pris en charge. Le client Amazon Redshift RSQL est pris en charge.
- Partitionnement de table (partitionnement par plage et par liste)
- Espaces de table
- Constraints

  - Unique
  - Clé étrangère
  - Clé primaire
  - Contraintes de validation
  - Contraintes d’exclusion

  Les contraintes d’unicité, de clé primaire et de clé étrangère sont autorisées, mais ne le sont qu’à titre informatif. Elles ne sont pas appliquées par le système, mais sont utilisées par le planificateur de requête.
- Héritage
- Colonnes système PostgreSQL

  Amazon Redshift SQL ne définit pas implicitement les colonnes système. Cependant, les noms de colonne système PostgreSQL suivants ne peuvent pas être utilisés en tant que noms de colonne définis par l’utilisateur : `oid`, `tableoid`, `xmin`, `cmin`, `xmax`, `cmax` et `ctid`. Pour de plus amples informations, veuillez consulter https://www.postgresql.org/docs/8.0/static/ddl-system-columns.html.
- Index
- Clause NULL dans les fonctions de fenêtrage
- Classements

  Amazon Redshift ne prend pas en charge les séquences de classement spécifiques à une région ou définies par l’utilisateur. Consultez Séquences de classement.
- Expressions de valeur
  - Expressions indicées
  - Constructeurs de tableau
  - Constructeurs de ligne
- Déclencheurs
- Gestion des données externes (SQL/MED)
- Fonctions de table
- Liste VALUES utilisée en tant que tables de constantes
- Séquences
- Recherche en texte intégral
- Les autorisations RULE et TRIGGER.

Amazon Redshift accorde ou révoque ces autorisations lorsque vous exécutez GRANT ALL ou REVOKE ALL, mais la présence ou l'absence des autorisations RULE et TRIGGER n'affecte en rien les autorisations d'accès du bénéficiaire.

Types de données PostgreSQL non pris en charge

En règle générale, si une requête essaie d'utiliser un type de données non pris en charge, y compris les conversions explicites ou implicites, elle retourne une erreur. Cependant, certaines requêtes qui utilisent les types de données non pris en charge s'exécutent sur le nœud principal, mais pas sur les nœuds de calcul. Consultez Fonctions SQL prises en charge sur le nœud principal.

Pour afficher la liste des types de données pris en charge, consultez Types de données.

Ces types de données PostgreSQL ne sont pas pris en charge par Amazon Redshift.

- Arrays (tableaux)
- BIT, BIT VARYING
- BYTEA
- Types composites
- Types énumérés
- Types géométriques (l'implémentation des types géométriques par Amazon Redshift est différente de celle de PostgreSQL)
- HSTORE
- JSON
- Types d'adresse réseau
- Types numériques
  - SERIAL, BIGSERIAL, SMALLSERIAL
  - MONEY
- Types d'identifiant d'objet
- Pseudo-types
- Types de plage
- Types de caractère spécial

- "char" – Un type interne à un octet (où le type de données nommé char est placé entre guillemets).
- name – Un type interne pour les noms d'objets.

Pour plus d'informations sur ces types, consultez Special Character Types dans la documentation PostgreSQL.

- Types de recherche de texte
- TXID_SNAPSHOT
- UUID
- xml

Fonctions PostgreSQL non prises en charge

La plupart des fonctions qui ne sont pas exclues ont une sémantique ou utilisation différente. Par exemple, certaines fonctions prises en charge ne s'exécutent que sur le nœud principal. De même, certaines fonctions non prises en charge pas ne renvoient pas une erreur lorsqu'elles s'exécutent sur le nœud principal. Le fait que ces fonctions ne renvoient pas d'erreur dans certains cas ne doit pas être considéré comme une indication que la fonction est prise en charge par Amazon Redshift.

> **Important**
>
> Ne présumez pas que la sémantique des éléments que Amazon Redshift et PostgreSQL ont en commun est identique. Veillez à consulter la rubrique Commandes SQL du Guide du développeur de bases de données Amazon Redshift pour comprendre les différences, souvent subtiles.

Pour de plus amples informations, veuillez consulter Fonctions SQL prises en charge sur le nœud principal.

Ces fonctions PostgreSQL ne sont pas prises en charge dans Amazon Redshift.

- Fonctions de demande de privilège d'accès
- Fonctions de verrou consultatif
- Fonctions d'agrégation
  - STRING_AGG()

  - ARRAY_AGG()
  - EVERY()
  - XML_AGG()
  - CORR()
  - COVAR_POP()
  - COVAR_SAMP()
  - REGR_AVGX(), REGR_AVGY()
  - REGR_COUNT()
  - REGR_INTERCEPT()
  - REGR_R2()
  - REGR_SLOPE()
  - REGR_SXX(), REGR_SXY(), REGR_SYY()
- Opérateurs et fonctions de tableau
- Fonctions de contrôle de sauvegarde
- Fonctions d’informations de commentaire
- Fonctions d’emplacement d’objet de base de données
- Fonctions de taille d’objet de base de données
- Fonctions et opérateurs Date/Time
  - CLOCK_TIMESTAMP()
  - JUSTIFY_DAYS(), JUSTIFY_HOURS(), JUSTIFY_INTERVAL()
  - PG_SLEEP()
  - TRANSACTION_TIMESTAMP()
- Fonctions de prise en charge d’ENUM
- Fonctions géométriques et opérateurs
- Fonctions d’accès fichier générique
- IS DISTINCT FROM
- Fonctions d’adresse réseau et opérateurs
- Fonctions mathématiques
  - DIV()
  - SETSEED()

  - WIDTH_BUCKET()
- Fonctions de retour d’ensembles
  - GENERATE_SERIES()
  - GENERATE_SUBSCRIPTS()
- Fonctions de plage et opérateurs
- Fonctions de contrôle de récupération
- Fonctions d’informations de récupération
- Fonction ROLLBACK TO SAVEPOINT
- Fonctions de demande de visibilité de schéma
- Fonctions de signalisation de serveur
- Fonctions de synchronisation d’instantané
- Fonctions de manipulation de séquence
- Fonctions de chaîne
  - BIT_LENGTH()
  - OVERLAY()
  - CONVERT(), CONVERT_FROM(), CONVERT_TO()
  - ENCODE()
  - FORMAT()
  - QUOTE_NULLABLE()
  - REGEXP_MATCHES()
  - REGEXP_SPLIT_TO_ARRAY()
  - REGEXP_SPLIT_TO_TABLE()
- Fonctions d’informations de catalogue système
- Fonctions d’informations système
  - CURRENT_CATALOG CURRENT_QUERY()
  - INET_CLIENT_ADDR()
  - INET_CLIENT_PORT()
  - INET_SERVER_ADDR() INET_SERVER_PORT()
  - PG_CONF_LOAD_TIME()
  - PG_IS_OTHER_TEMP_SCHEMA()

  - PG_LISTENING_CHANNELS()
  - PG_MY_TEMP_SCHEMA()
  - PG_POSTMASTER_START_TIME()
  - PG_TRIGGER_DEPTH()
  - SHOW VERSION()
- Fonctions de recherche de texte et opérateurs
- Fonctions de transaction IDs et d'instantanés
- Fonctions déclencheur
- Fonctions XML

Utilisation de SQL

Rubriques

- Conventions du guide de référence SQL
- Éléments de base
- Expressions
- Conditions

Le langage SQL se compose de commandes et de fonctions que vous utilisez pour travailler avec les bases de données et les objets de base de données. Il applique également les règles concernant l'utilisation des types de données, les expressions et les littéraux.

Conventions du guide de référence SQL

Cette section explique les conventions utilisées pour écrire la syntaxe des expressions, commandes et fonctions SQL décrites dans la section de référence SQL.

Caractère	Description
CAPS	Les mots en lettres majuscules sont des mots clés.
[ ]	Les crochets indiquent des arguments facultatifs. Plusieurs arguments entre crochets signifient que vous pouvez choisir n'importe quel nombre d'argumen ts. En outre, les arguments entre parenthèses sur des lignes séparées

Caractère	Description
	indiquent que l'analyseur Amazon Redshift s'attend à ce que les arguments soient dans l'ordre où ils sont énumérés dans la syntaxe. Pour obtenir un exemple, consultez SELECT.
{ }	Les accolades indiquent que vous devez choisir l'un des arguments proposés.
|	Les barres verticales indiquent que vous pouvez choisir entre les arguments.
italique	Les mots en italique correspondent à des espaces réservés. Vous devez insérer la valeur appropriée à la place du mot en italique.
. . .	Les trois points de suspension indiquent que vous pouvez répéter l'élément précédent.
'	Les mots entre apostrophes droites signifient que vous devez taper les apostrophes.

Éléments de base

Rubriques

- Noms et identificateurs
- Littéraux
- Valeurs null
- Types de données
- Séquences de classement

Cette section concerne les règles d'utilisation des noms d'objet de base de données, des littéraux, des valeurs null et des types de données.

Noms et identificateurs

Les noms identifient les objets de base de données, y compris les tables et les colonnes, ainsi que les utilisateurs et les mots de passe. Les termes nom et identificateur peuvent être utilisés indifféremment. Il existe deux types d'identificateurs, les identificateurs standard et les identificateurs

entre guillemets ou délimités. Les identificateurs doivent comporter uniquement des caractères imprimables UTF-8. Les lettres ASCII des identificateurs standard et délimités sont insensibles à la casse et sont converties en minuscules dans la base de données. Dans les résultats des requêtes, les noms de colonne sont retournés en minuscules par défaut. Pour retourner les noms de colonne en majuscules, définissez le paramètre de configuration describe_field_name_in_uppercase sur la valeur **`true`**.

Identificateurs standard

Les identificateurs SQL standard se conforment à un ensemble de règles et doivent :

- Commencer par un caractère alphabétique ou de soulignement ASCII codé sur un octet, ou un caractère multioctet UTF-8 d'une longueur de deux ou quatre octets.
- Les caractères suivants peuvent être des caractères alphanumériques ou de soulignement ASCII codés sur un octet, des symboles dollar, ou des caractères multioctets UTF-8 d'une longueur de deux ou quatre octets.
- Avoir une longueur comprise entre 1 et 127 octets, sans les guillemets pour les identifiants délimités.
- Ne contenir ni guillemets ni espaces.
- Ne pas être un mot réservé SQL.

Identificateurs délimités

Les identificateurs délimités (aussi appelés identificateurs entre guillemets) commencent et se terminent par des guillemets ("). Si vous utilisez un identificateur délimité, vous devez utiliser les guillemets pour chaque référence à cet objet. L'identifiant peut contenir tous les caractères imprimables UTF-8 standard autres que les guillemets droits eux-mêmes. Par conséquent, vous pouvez créer des noms de colonne ou de table qui incluent tout caractère autre que les caractères illégaux, tels que les espaces ou le symbole pourcentage (%).

Les lettres ASCII des identificateurs délimités sont insensibles à la casse et sont converties en minuscules. Pour utiliser des guillemets droits dans une chaîne de caractères, vous devez les faire précéder d'un autre caractère de guillemets droits.

Identifiants sensibles à la casse

Les identifiants sensibles à la casse (également appelés identifiants en casse mixte) peuvent contenir des lettres majuscules et minuscules. Pour utiliser des identifiants sensibles à la casse, vous pouvez

définir la configuration `enable_case_sensitive_identifier` sur `true`. Vous pouvez définir cette configuration pour le cluster ou une session. Pour plus d'informations, consultez Valeurs des paramètres par défaut dans le Guide de gestion Amazon Redshift et enable_case_sensitive_identifier.

Nom des colonnes système

Les noms de colonnes système PostgreSQL suivants ne peuvent pas être utilisés en tant que noms de colonne définis par l'utilisateur. Pour de plus amples informations, veuillez consulter https://www.postgresql.org/docs/8.0/static/ddl-system-columns.html.

- `oid`
- `tableoid`
- `xmin`
- `cmin`
- `xmax`
- `cmax`
- `ctid`

Exemples

Ce tableau présente des exemples d'identificateurs délimités, le résultat obtenu et une explication :

Syntaxe	Résultat	Explication
"group"	groupe	GROUP est un mot réservé, son utilisation dans un identifiant nécessite donc des guillemets.
"""WHERE"""	"where"	WHERE est également un mot réservé. Pour inclure des guillemets droits dans la chaîne, échappez chaque caractère de guillemets droits par d'autres caractères de guillemets droits.
"This name"	this name	Les guillemets droits sont nécessaires pour préserver l'espace.
"This ""IS IT"""	"This ""IS IT"""	Les guillemets entourant IS IT doivent être précédés d'un guillemet supplémentaire pour faire partie du nom.

Pour créer un groupe nommé table avec une colonne nommée this "is it" :

```
create table "group" (
"This ""IS IT""" char(10));
```

Les requêtes suivantes retournent le même résultat :

```
select "This ""IS IT"""
from "group";

this "is it"
--------------
(0 rows)
```

```
select "this ""is it"""
from "group";

this "is it"
--------------
(0 rows)
```

La syntaxe complète suivante `table.column` renvoie aussi le même résultat :

```
select "group"."this ""is it"""
from "group";

this "is it"
--------------
(0 rows)
```

La commande CREATE TABLE suivante crée une table avec une barre oblique dans un nom de colonne :

```
create table if not exists city_slash_id(
                  "city/id" integer not null,
                  state char(2) not null);
```

Littéraux

Un littéral ou une constante est une valeur de données fixe, composée d'une séquence de caractères ou d'une constante numérique. Amazon Redshift prend en charge plusieurs types de littéraux, notamment :

- Les littéraux numériques pour les entiers, les décimaux et les nombres à virgule flottante. Pour de plus amples informations, veuillez consulter Littéraux entiers et à virgule flottante.
- Les littéraux caractère, également appelés chaînes, chaînes de caractères ou constantes caractère
- Les littéraux de type datetime et interval utilisés avec les types de données datetime. Pour plus d'informations, consultez Littéraux de type date, heure et horodatage et Types de données d'intervalle et littéraux.

Valeurs null

Si une colonne d'une ligne est manquante, inconnue ou non applicable, sa valeur est null ou est dite contenir une valeur null. Les valeurs null peuvent apparaître dans les champs de n'importe quel type de données qui n'est pas limité par les contraintes de clé primaire ou NOT NULL. Une valeur null n'est pas équivalente à la valeur zéro ou à une chaîne vide.

Toute expression arithmétique contenant une valeur null est toujours analysée comme null. Tous les opérateurs à l'exception de la concaténation retournent une valeur null en cas d'argument ou d'opérande null.

Pour tester les valeurs null, utilisez les conditions de comparaison IS NULL et IS NOT NULL. Comme null représente une absence de données, une valeur null n'est pas égale à une autre valeur, null ou pas.

Types de données

Rubriques

- Caractères multioctets
- Types numériques
- Types caractères
- Types datetime
- Type Boolean
- Type HLLSKETCH

- Type SUPER
- Type VARBYTE
- Compatibilité et conversion de types

Chaque valeur qu'Amazon Redshift stocke ou extrait possède un type de données avec un ensemble fixe de propriétés associées. Les types de données sont déclarés lorsque les tables sont créées. Un type de données contraint l'ensemble des valeurs qu'une colonne ou un argument peut contenir.

Le tableau suivant répertorie les types de données que vous pouvez utiliser dans les tables Amazon Redshift.

Type de données	Alias	Description
SMALLINT	INT2	Entier signé sur deux octets
INTEGER	ENTIER, INT4	Entier signé sur quatre octets
BIGINT	INT8	Entier signé sur huit octets
DECIMAL	NUMERIC	Valeur numérique exacte avec précision sélectionnable
REAL	FLOAT4	Nombre à virgule flottante simple précision
DOUBLE PRECISION	FLOAT8, FLOTTEUR	Nombre à virgule flottante de double précision
CHAR	CHARACTER, NCHAR, BPCHAR	Chaîne de caractères de longueur fixe
VARCHAR	CHARACTER VARYING, NVARCHAR, TEXT	Chaîne de caractères de longueur variable avec une limite définie par l'utilisateur
DATE		Date calendaire (année, mois, jour)
TIME	TIME WITHOUT TIME ZONE	Time of day

Type de données	Alias	Description
TIMETZ	TIME WITH TIME ZONE	Time of day with time zone
TIMESTAMP	TIMESTAMP WITHOUT TIME ZONE	Date et heure (sans fuseau horaire)
TIMESTAMPTZ	TIMESTAMP WITH TIME ZONE	Date et heure (avec fuseau horaire)
INTERVALLE D'UNE ANNÉE À L'AUTRE		Durée en ordre annuel par mois
INTERVALLE D'UN JOUR À L'AUTRE		Durée en jours jusqu'au deuxième ordre
BOOLEAN	BOOL	Booléen logique (true/false)
HLLSKETCH		Type utilisé avec les HyperLogLog croquis.
SUPER		Type de données superset qui englobe tous les types scalaires d’Amazon Redshift, y compris les types complexes tels que ARRAY et STRUCTS.
VARBYTE	VARBINARY, BINARY VARYING	Valeur binaire de longueur variable
GEOMETRY		Données spatiales
GEOGRAPHY		Données spatiales

> **Note**
>
> Pour plus d'informations sur les types de données non pris en charge, tels que "char" (notez que char est placé entre guillemets), consultez Types de données PostgreSQL non pris en charge.

Caractères multioctets

Le type de données VARCHAR prend en charge les caractères multioctets UTF-8 jusqu'à un maximum de quatre octets. Les caractères de cinq octets ou plus ne sont pas pris en charge. Pour calculer la taille d'une colonne VARCHAR qui contient des caractères multioctets, multipliez le nombre de caractères par le nombre d'octets par caractère. Par exemple, si une chaîne possède quatre caractères chinois et que chaque caractère est long de trois octets, vous avez besoin d'une colonne VARCHAR(12) pour stocker la chaîne.

Le type de données VARCHAR ne prend pas en charge les points de code UTF-8 non valides suivants :

`0xD800 – 0xDFFF` (Séquences d'octets :`ED A0 80` – `ED BF BF`)

Le type de données CHAR ne prend pas en charge les caractères multioctets.

Types numériques

Rubriques

- Types d'entier
- Type DECIMAL ou NUMERIC
- Notes sur l'utilisation des colonnes DECIMAL ou NUMERIC 128 bits
- Types à virgule flottante
- Calculs avec les valeurs numériques
- Littéraux entiers et à virgule flottante
- Exemples avec les types numériques

Les types de données numériques incluent les entiers, les décimaux et les nombres à virgule flottante.

Types d’entier

Utilisez les types de données SMALLINT, INTEGER et BIGINTpour stocker les nombres entiers de différentes plages. Vous ne pouvez pas stocker de valeurs en dehors de la plage autorisée pour chaque type.

Nom	Stockage	Range
SMALLINT ou INT2	2 bytes	-32768 à +32767
INTEGER, INT ou INT4	4 bytes	-2147483648 à +2147483647
BIGINT ou INT8	8 bytes	-9223372036854775808 à 9223372036854775807

Type DECIMAL ou NUMERIC

Utilisez le type de données DECIMAL ou NUMERIC pour stocker les valeurs avec une précision définie par l’utilisateur. Les mots clés DECIMAL et NUMERIC sont interchangeables. Dans ce document, decimal est le terme privilégié pour ce type de données. Le terme numeric (numérique) est utilisé de façon générique pour faire référence aux types de données integer, decimal et floating-point (entier, décimal et virgule flottante).

Stockage	Range
Variable, jusqu’à 128 bits pour les types DECIMAL non compressés.	Entiers signés 128 bits avec précision maximale de 38 chiffres.

Définissez une colonne DECIMAL dans une table en spécifiant une precision (précision) et une scale (échelle) :

```
decimal(precision, scale)
```

precision

Le nombre total de chiffres significatifs dans la valeur entière : le nombre de chiffres de chaque côté de la virgule. Par exemple, le nombre `48.2891` a une précision de 6 et une échelle de 4. La précision par défaut, si elle n'est pas spécifiée, est de 18. La précision maximale est de 38.

Si le nombre de chiffres à gauche de la virgule dans une valeur d'entrée dépasse la précision de la colonne moins son échelle, la valeur ne peut pas être copiée dans la colonne (ou insérée ou mise à jour). Cette règle s'applique à toute valeur qui se trouve en dehors de la plage de la définition de la colonne. Par exemple, la plage autorisée de valeurs pour une colonne `numeric(5,2)` s'étend de `-999.99` à `999.99`.

échelle

Le nombre de chiffres décimaux de la partie fractionnaire de la valeur, à droite de la virgule. Les entiers possèdent une échelle égale à zéro. Dans une spécification de colonne, la valeur de l'échelle doit être inférieure ou égale à la valeur de la précision. L'échelle par défaut, si elle n'est pas spécifiée, est de 0. L'échelle maximale est de 37.

Si l'échelle d'une valeur d'entrée chargée dans une table est supérieure à l'échelle de la colonne, la valeur est arrondie à l'échelle spécifiée. Par exemple, la colonne PRICEPAID de la table SALES est une colonne DECIMAL(8,2). Si une valeur DECIMAL(8,4) est insérée dans la colonne PRICEPAID, la valeur est arrondie à une échelle de 2.

```
insert into sales
values (0, 8, 1, 1, 2000, 14, 5, 4323.8951, 11.00, null);

select pricepaid, salesid from sales where salesid=0;

pricepaid | salesid
-----------+---------
4323.90 |       0
(1 row)
```

Cependant, les résultats de conversions explicites de valeurs sélectionnées dans les tables ne sont pas arrondis.

> **Note**
>
> La valeur positive maximale que vous pouvez insérer dans une colonne DECIMAL(19,0) est `9223372036854775807` (2^{63} -1). La valeur négative maximale est `-9223372036854775808`. Par exemple, une tentative d'insérer la valeur `9999999999999999999` (19 fois le chiffre neuf) entraîne une erreur de dépassement de capacité. Quelque soit le placement de la virgule décimale, la plus grande chaîne qu'Amazon Redshift puisse représenter comme nombre DECIMAL est `9223372036854775807`.
> Par exemple, la plus grande valeur que vous puissiez charger dans une colonne DECIMAL(19,18) est `9.223372036854775807`.
> Ces règles sont dues au fait que les valeurs de type DECIMAL d'une précision de 19 chiffres significatifs ou moins sont stockées en interne sous forme d'entiers de 8 octets, tandis que celles de type DECIMAL d'une précision de 20 à 38 chiffres significatifs sont stockées sous forme d'entiers de 16 octets.

Notes sur l'utilisation des colonnes DECIMAL ou NUMERIC 128 bits

N'attribuez pas de façon arbitraire une précision maximale aux colonnes DECIMAL, sauf si vous avez la certitude que votre application a besoin de cette précision. Les valeurs 128 bits utilisent deux fois plus d'espace disque que les valeurs 64 bits et peuvent ralentir le temps d'exécution des requêtes.

Types à virgule flottante

Utilisez les types de données REAL et DOUBLE PRECISION pour stocker les valeurs numériques avec une précision variable. Ces types sont des types inexacts, ce qui signifie que certaines valeurs sont stockées comme approximations, de telle sorte que le stockage et le retour d'une valeur spécifique peuvent se traduire par de légers écarts. Si vous avez besoin d'un stockage et d'un calcul exacts (pour des montants monétaires, par exemple), utilisez le type de données DECIMAL.

REAL représente le format à virgule flottante simple précision, conformément à la norme IEEE 754 pour l'arithmétique binaire à virgule flottante. Il a une précision d'environ 6 chiffres et une plage d'environ 1E-37 à 1E+37. Vous pouvez également spécifier ce type de données sous la forme FLOAT4.

DOUBLE PRECISION représente le format de virgule flottante en double précision, conformément à la norme IEEE 754 pour l'arithmétique binaire en virgule flottante. Il a une précision d'environ 15 chiffres et une plage d'environ 1E-307 à 1E+308. Vous pouvez également spécifier ce type de données comme FLOAT ou FLOAT8.

Outre les valeurs numériques ordinaires, les types à virgule flottante possèdent plusieurs valeurs spéciales. Quand vous utilisez ces valeurs dans SQL, entourez-les de guillemets simples :

- `NaN` – not-a-number
- `Infinity` : infini
- `-Infinity` : infini négatif

Par exemple, pour insérer not-a-number dans une colonne `day_charge` de table, `customer_activity` exécutez le code SQL suivant :

```
insert into customer_activity(day_charge) values('NaN');
```

Calculs avec les valeurs numériques

Dans ce contexte, le terme calcul fait référence aux opérations mathématiques binaires : addition, soustraction, multiplication et division. Cette section décrit les types de retour attendus pour ces opérations, ainsi que la formule spécifique appliquée pour déterminer la précision et l'échelle lorsque les types de données DECIMAL sont impliqués.

Lorsque des valeurs numériques sont calculées pendant le traitement des requêtes, vous pouvez rencontrer des cas où le calcul est impossible et où la requête renvoie une erreur de dépassement de capacité numérique. Vous pouvez également rencontrer des cas où l'échelle des valeurs calculées varie ou est inattendue. Pour certaines opérations, vous pouvez utiliser un transtypage explicite (promotion de type) ou des paramètres de configuration Amazon Redshift pour contourner ces problèmes.

Pour plus d'informations sur les résultats de calculs similaires avec les fonctions SQL, consultez Fonctions d'agrégation.

Types de retour pour les calculs

Compte tenu de l'ensemble des types de données numériques pris en charge par Amazon Redshift, le tableau suivant présente les types de retour attendus pour les opérations d'addition, de soustraction, de multiplication et de division. La première colonne sur la gauche du tableau représente le premier opérande dans le calcul et la ligne du haut le second opérande.

Opérande 1	Opérande 2	Type de retour
INT2	INT2	INT2
INT2	INT4	INT4
INT2	INT8	INT8
INT2	DECIMAL	DECIMAL
INT2	FLOAT4	FLOAT8
INT2	FLOAT8	FLOAT8
INT4	INT4	INT4
INT4	INT8	INT8
INT4	DECIMAL	DECIMAL
INT4	FLOAT4	FLOAT8
INT4	FLOAT8	FLOAT8
INT8	INT8	INT8
INT8	DECIMAL	DECIMAL
INT8	FLOAT4	FLOAT8
INT8	FLOAT8	FLOAT8
DECIMAL	DECIMAL	DECIMAL
DECIMAL	FLOAT4	FLOAT8
DECIMAL	FLOAT8	FLOAT8
FLOAT4	FLOAT8	FLOAT8
FLOAT8	FLOAT8	FLOAT8

Précision et échelle des résultats DECIMAL calculés

Le tableau suivant résume les règles de calcul de la précision et de l'échelle obtenues lorsque les opérations mathématiques retournent des résultats DECIMAL. Dans cette table, `p1` et `s1` représentent la précision et l'échelle du première opérande d'un calcul, tandis que `p2` et `s2` représentent la précision et l'échelle du second opérande. (Quels que soient les calculs, la précision maximale du résultat est de 38 et l'échelle maximale du résultat de 38 également.)

Opération	Précision et échelle du résultat
+ ou -	Évolutivité = `max(s1,s2)` Précision = `max(p1-s1,p2-s2)+1+scale`
*	Évolutivité = `s1+s2` Précision = `p1+p2+1`
/	Évolutivité = `max(4,s1+p2-s2+1)` Précision = `p1-s1+ s2+scale`

Par exemple, les colonnes PRICEPAID et COMMISSION de la table SALES sont toutes deux des colonnes DECIMAL(8,2). Si vous divisez PRICEPAID par COMMISSION (ou inversement), la formule est appliquée comme suit :

```
Precision = 8-2 + 2 + max(4,2+8-2+1)
= 6 + 2 + 9 = 17

Scale = max(4,2+8-2+1) = 9

Result = DECIMAL(17,9)
```

Le calcul suivant constitue la règle générale pour le calcul de la précision et de l'échelle obtenues dans le cas des opérations effectuées sur les valeurs DECIMAL avec les opérateurs définis tels que UNION, INTERSECT et EXCEPT, ou les fonctions comme COALESCE et DECODE :

```
Scale = max(s1,s2)
Precision = min(max(p1-s1,p2-s2)+scale,19)
```

Par exemple, une DEC1 table avec une colonne DECIMAL (7,2) est jointe à une DEC2 table avec une colonne DECIMAL (15,3) pour créer une table. DEC3 Le schéma de DEC3 montre que la colonne devient une colonne NUMERIC (15,3).

```
create table dec3 as select * from dec1 union select * from dec2;
```

Résultat

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'dec3';

column |     type      | encoding | distkey | sortkey
-------+---------------+----------+---------+---------
c1     | numeric(15,3) | none     | f       | 0
```

Dans l'exemple ci-dessus, la formule est appliquée comme suit :

```
Precision = min(max(7-2,15-3) + max(2,3), 19)
= 12 + 3 = 15

Scale = max(2,3) = 3

Result = DECIMAL(15,3)
```

Remarques sur les opérations de division

Pour les opérations de division, divide-by-zero les conditions renvoient des erreurs.

La limite d'échelle de 100 est appliquée après le calcul de la précision et de l'échelle. Si l'échelle de résultat calculée est supérieure à 100, les résultats de la division sont mis à l'échelle comme suit :

- Précision = `precision - (scale - max_scale)`
- Évolutivité = `max_scale`

Si la précision calculée est supérieure à la précision maximale (38), la précision est réduite à 38, et l'échelle devient le résultat de : `max((38 + scale - precision), min(4, 100))`

Conditions de dépassement de capacité

Le dépassement de capacité est contrôlé pour tous les calculs numériques. Les données DECIMAL avec une précision de 19 ou moins sont stockées en tant qu'entiers 64 bits. Les données DECIMAL avec une précision supérieure à 19 sont stockées sous forme d'entiers 128 bits. La précision maximale de toutes les valeurs DECIMAL est 38 et l'échelle maximale 37. Les erreurs de dépassement de capacité se produisent quand une valeur dépasse ces limites, qui s'appliquent aux jeux de résultats intermédiaires et finaux :

- Le transtypage explicite se traduit par des erreurs de dépassement de capacité à l'exécution lorsque les valeurs de données spécifiques ne correspondent pas à la précision ou à l'échelle spécifiée par la fonction cast. Par exemple, vous ne pouvez pas effectuer une conversion de type de toutes les valeurs de la colonne PRICEPAID de la table SALES (une colonne DECIMAL(8,2)) et retourner un résultat DECIMAL(7,3) :

  ```
  select pricepaid::decimal(7,3) from sales;
  ERROR:  Numeric data overflow (result precision)
  ```

  Cette erreur se produit, parce que certaines des valeurs les plus grandes de la colonne PRICEPAID ne peuvent pas être converties.
- Les opérations de multiplication produisent des résultats dans lesquels l'échelle du résultat est la somme des échelles de chaque opérande. Si les deux opérandes ont une échelle de 4, par exemple, l'échelle du résultat est 8, ce qui ne laisse que 10 chiffres à gauche de la virgule. Par conséquent, il est relativement facile de se trouver en situation de dépassement de capacité lors de la multiplication de deux grands nombres ayant une échelle significative.

  L'exemple suivant entraîne une erreur de dépassement de capacité.

  ```
  SELECT CAST(1 AS DECIMAL(38, 20)) * CAST(10 AS DECIMAL(38, 20));
  ERROR: 128 bit numeric data overflow (multiplication)
  ```

  Vous pouvez contourner l'erreur de dépassement de capacité en utilisant la division au lieu de la multiplication. Utilisez l'exemple suivant pour diviser par 1 divisé par le diviseur d'origine.

  ```
  SELECT CAST(1 AS DECIMAL(38, 20)) / (1 / CAST(10 AS DECIMAL(38, 20)));
  +----------+
  | ?column? |
  +----------+
  | 10       |
  ```

```
+----------+
```

Calculs numériques avec les types INTEGER et DECIMAL

Lorsqu'un des opérandes d'un calcul est de type de données INTEGER et que l'autre opérande est DECIMAL, l'opérande INTEGER est implicitement converti en DECIMAL :

- INT2 (SMALLINT) est converti en DECIMAL (5,0)
- INT4 (INTEGER) est converti en DECIMAL (10,0)
- INT8 (BIGINT) est converti en DECIMAL (19,0)

Par exemple, si vous multipliez SALES.COMMISSION, colonne DECIMAL(8,2) et SALES.QTYSOLD, colonne SMALLINT, le calcul est converti comme suit :

```
DECIMAL(8,2) * DECIMAL(5,0)
```

Littéraux entiers et à virgule flottante

Les littéraux ou constantes qui représentent des nombres peuvent être des entiers ou à virgule flottante.

Littéraux entiers

Une constante entière est une séquence des chiffres 0-9, avec un signe positif (+) ou négatif (-) facultatif devant les chiffres.

Syntaxe

```
[ + | - ] digit ...
```

Exemples

Les entiers valides incluent les éléments suivants :

```
23
-555
+17
```

Littéraux à virgule flottante

Les littéraux à virgule flottante (également appelés littéraux décimaux, numériques ou fractionnaires) sont des séquences de chiffres qui peuvent inclure une virgule décimale et, le cas échéant, le symbole exposant (e).

Syntaxe

```
[ + | - ] digit ... [ . ] [ digit ...]
[ e | E [ + | - ] digit ... ]
```

Arguments

e | E

e ou E indique que le nombre est spécifié en notation scientifique.

Exemples

Les littéraux à virgule flottante incluent les éléments suivants :

```
3.14159
-37.
2.0e19
-2E-19
```

Exemples avec les types numériques

Instruction CREATE TABLE

L'instruction CREATE TABLE suivante illustre la déclaration de différents types de données numériques :

```
create table film (
film_id integer,
language_id smallint,
original_language_id smallint,
rental_duration smallint default 3,
rental_rate numeric(4,2) default 4.99,
length smallint,
replacement_cost real default 25.00);
```

Tentative d'insérer un entier qui est hors de portée

L'exemple suivant tente d'insérer la valeur 33 000 dans une colonne SMALLINT.

```
insert into film(language_id) values(33000);
```

La plage valide pour SMALLINT étant -32768 à +32767, Amazon Redshift retourne une erreur.

```
An error occurred when executing the SQL command:
insert into film(language_id) values(33000)

ERROR: smallint out of range [SQL State=22003]
```

Insérer une valeur décimale dans une colonne de type entier

L'exemple suivant insère la valeur décimale a dans une colonne INT.

```
insert into film(language_id) values(1.5);
```

Cette valeur est insérée, mais arrondie à la valeur entière 2.

Insertion réussie d'une valeur décimale parce que son échelle est arrondie

L'exemple suivant insère une valeur décimale ayant une plus grande précision que la colonne.

```
insert into film(rental_rate) values(35.512);
```

Dans ce cas, la valeur `35.51` est insérée dans la colonne.

Tentative d'insertion d'une valeur décimale qui est hors de portée

Dans ce cas, la valeur `350.10` est hors de portée. Le nombre de chiffres pour les valeurs de colonnes DECIMAL est égal à la précision de la colonne moins son échelle (4 moins 2 pour la colonne RENTAL_RATE). Autrement dit, la plage autorisée pour une colonne `DECIMAL(4,2)` s'étend de `-99.99` à `99.99`.

```
insert into film(rental_rate) values (350.10);
ERROR:  numeric field overflow
DETAIL:  The absolute value is greater than or equal to 10^2 for field with precision
 4, scale 2.
```

Insérer des valeurs de précision variable dans une colonne REAL

L'exemple suivant insère des valeurs de précision variable dans une colonne REAL.

```
insert into film(replacement_cost) values(1999999.99);

insert into film(replacement_cost) values(1999.99);

select replacement_cost from film;

+------------------+
| replacement_cost |
+------------------+
| 2000000          |
| 1999.99          |
+------------------+
```

La valeur `1999999.99` est convertie en `2000000` pour satisfaire aux exigences de précision pour la colonne `REAL`. La valeur `1999.99` est chargée en l'état.

Types caractères

Rubriques

- Stockage et plages
- CHAR ou CHARACTER
- VARCHAR ou CHARACTER VARYING
- Types NCHAR et NVARCHAR
- Types TEXT et BPCHAR
- Signification des blancs de fin
- Exemples avec les types caractères

Les types de données caractères incluent CHAR (caractère) et VARCHAR (caractère variable).

Stockage et plages

Les types de données CHAR et VARCHAR sont définis en termes d'octets, pas de caractères. Comme une colonne CHAR ne peut contenir que des caractères d'un octet, une colonne CHAR(10) peut contenir une chaîne d'une longueur maximale de 10 octets. Une donnée VARCHAR peut

contenir des caractères multioctets, jusqu'à un maximum de quatre octets par caractère. Par exemple, une colonne VARCHAR(12) peut contenir 12 caractères codés sur un octet, 6 caractères codés sur deux octets, 4 caractères codés sur trois octets ou 3 caractères codés sur quatre octets.

Nom	Stockage	Plage (largeur de colonne)
CHAR, CHARACTER ou NCHAR	Longueur de la chaîne, blancs de fin inclus (le cas échéant)	4096 bytes
VARCHAR, CHARACTER VARYING ou NVARCHAR	4 octets + le nombre total d'octets des caractères, où chaque caractère peut être codé sur 1 à 4 octets.	65535 octets (64 K -1)
BPCHAR	Converti en longueur fixe CHAR(256).	256 bytes
TEXT	Converti en VARCHAR(256).	260 bytes

> **Note**
>
> La syntaxe CREATE TABLE prend en charge le mot clé MAX pour les types de données character. Par exemple :
>
> ```
> create table test(col1 varchar(max));
> ```
>
> Le paramètre MAX définit la largeur de la colonne en tant que 4096 octets pour CHAR ou 65535 octets pour VARCHAR.

CHAR ou CHARACTER

Utilisez une colonne CHAR ou CHARACTER pour stocker les chaînes de longueur fixe. Ces chaînes étant remplies de blancs, une colonne CHAR(10) occupe toujours 10 octets de stockage.

```
char(10)
```

Une colonne CHAR sans spécification de longueur se traduit par une colonne CHAR(1).

VARCHAR ou CHARACTER VARYING

Utilisez une colonne VARCHAR ou CHARACTER VARYING pour stocker des chaînes de longueur variable avec une limite fixe. Comme ces chaînes ne sont pas remplies avec des blancs, une colonne VARCHAR(120) se compose d'un maximum de 120 caractères codés sur un octet, de 60 caractères codés sur deux octets, de 40 caractères codés sur trois octets ou de 30 caractères codés sur quatre octets.

```
varchar(120)
```

Si vous utilisez le type de données VARCHAR sans spécification de la longueur dans une instruction CREATE TABLE, la longueur par défaut est 256. Utilisée dans une expression, la taille de la sortie est déterminée à l'aide de l'expression d'entrée (jusqu'à 65535).

Types NCHAR et NVARCHAR

Vous pouvez créer des colonnes avec les types NCHAR et NVARCHAR (aussi appelés types NATIONAL CHARACTER et NATIONAL CHARACTER VARYING). Ces types sont convertis en types CHAR et VARCHAR, respectivement, et sont stockés dans le nombre d'octets spécifié.

Une colonne NCHAR sans spécification de longueur est convertie en colonne CHAR(1).

Une colonne NVARCHAR sans spécification de longueur est convertie en colonne VARCHAR(256).

Types TEXT et BPCHAR

Vous pouvez créer une table Amazon Redshift avec une colonne TEXT, mais celle-ci est convertie en une colonne VARCHAR(256) qui accepte des valeurs de longueur variable avec un maximum de 256 caractères.

Vous pouvez créer une colonne Amazon Redshift avec un type BPCHAR (blank-padded character), qu'Amazon Redshift convertit en une colonne CHAR(256) de longueur fixe.

Signification des blancs de fin

Les types de données CHAR et VARCHAR stockent les chaînes de longueur maximale de n octets. Une tentative de stocker une chaîne plus longue en une colonne de l'un de ces types entraîne une erreur, sauf si les caractères supplémentaires sont tous des espaces (des blancs), auquel cas la chaîne est tronquée à la longueur maximale. Si la chaîne est plus courte que la longueur maximale, les valeurs CHAR sont remplies de blancs, mais les valeurs VARCHAR stockent la chaîne sans blancs.

Les blancs de fin des valeurs CHAR sont toujours insignifiants sur le plan sémantique. Ils sont ignorés lorsque vous comparez deux valeurs CHAR, ne sont pas inclus dans les calculs LENGTH et sont supprimés lorsque vous convertissez une valeur CHAR en un autre type de chaîne.

Les espaces de fin des valeurs VARCHAR et CHAR sont traités comme sans importance du point de vue sémantique lorsque les valeurs sont comparées.

Les longueurs de calcul retournent la longueur des chaînes de caractères VARCHAR avec les espaces de fin inclus dans la longueur. Les blancs de fin ne comptent pas dans la longueur des chaînes de caractères de longueur fixe.

Exemples avec les types caractères

Instruction CREATE TABLE

L'instruction CREATE TABLE suivante illustre l'utilisation de types de données VARCHAR et CHAR :

```
create table address(
address_id integer,
address1 varchar(100),
address2 varchar(50),
district varchar(20),
city_name char(20),
state char(2),
postal_code char(5)
);
```

Les exemples suivants s'appuient sur cette table.

Blancs de fin dans les chaînes de caractères de longueur variable

Comme il ADDRESS1 s'agit d'une colonne VARCHAR, les derniers blancs de la deuxième adresse insérée sont sémantiquement insignifiants. En d’autres termes, les deux adresses insérées correspondent.

```
insert into address(address1) values('9516 Magnolia Boulevard');

insert into address(address1) values('9516 Magnolia Boulevard  ');
```

```
select count(*) from address
where address1='9516 Magnolia Boulevard';

count
-------
2
(1 row)
```

Si la ADDRESS1 colonne était une colonne CHAR et que les mêmes valeurs étaient insérées, la requête COUNT (*) reconnaîtrait les chaînes de caractères comme étant identiques et renverrait les résultats2.

Résultats de la fonction LENGTH

La fonction LENGTH reconnaît les espaces de fin dans les colonnes VARCHAR :

```
select length(address1) from address;

length
--------
23
25
(2 rows)
```

La valeur `Augusta` dans la colonne CITY_NAME, qui est une colonne CHAR, renvoie toujours une longueur de 7 caractères, indépendamment des espaces de fin de la chaîne en entrée.

Valeurs qui dépassent la longueur de la colonne

Les chaînes de caractères ne sont pas tronquées pour s’adapter à la largeur déclarée de la colonne :

```
insert into address(city_name) values('City of South San Francisco');
ERROR: value too long for type character(20)
```

Une solution pour contourner ce problème consiste à convertir la valeur en la taille de la colonne :

```
insert into address(city_name)
values('City of South San Francisco'::char(20));
```

Dans ce cas, les 20 premiers caractères de la chaîne (`City of South San Fr`) sont chargés dans la colonne.

Types datetime

Rubriques

- Stockage et plages
- DATE
- TIME
- TIMETZ
- TIMESTAMP
- TIMESTAMPTZ
- Exemples avec les types datetime
- Littéraux de type date, heure et horodatage
- Types de données d'intervalle et littéraux

Les types de données datetime incluent DATE, TIMESTAMP et TIMESTAMPTZ.

Stockage et plages

Nom	Stockage	Range	Résolution
DATE	4 bytes	4713 av. J.-C. à 294276 apr. J.-C.	1 jour
TIME	8 bytes	De 00:00:00 à 24:00:00	1 microseconde
TIMETZ	8 bytes	De 00:00:00+1459 à 00:00:00+1459	1 microseconde

Nom	Stockage	Range	Résolution
TIMESTAMP	8 bytes	4713 av. J.-C. à 294276 apr. J.-C.	1 microseconde
TIMESTAMP TZ	8 bytes	4713 av. J.-C. à 294276 apr. J.-C.	1 microseconde

DATE

Utilisez le type de données DATE pour stocker les dates calendaires simples sans horodatage.

TIME

TIME est un alias de TIME WITHOUT TIME ZONE.

Utilisez le type de données TIME pour stocker l'heure de la journée.

Les colonnes TIME stockent des valeurs avec un maximum de six digits de précision pour les secondes fractionnées.

Par défaut, les valeurs TIME sont en temps universel coordonné (UTC) dans les tables utilisateur et les tables système Amazon Redshift.

TIMETZ

TIMETZ est un alias de TIMES WITH TIME ZONE.

Utilisez le type de données TIMETZ pour stocker l'heure de la journée avec un fuseau horaire.

Les colonnes TIMETZ stockent des valeurs avec un maximum de six digits de précision pour les secondes fractionnées.

Par défaut, les valeurs TIPZ sont UTC dans les tables utilisateur et dans les tables système Amazon Redshift.

TIMESTAMP

TIMESTAMP est un alias de TIMESTAMP WITHOUT TIME ZONE.

Utilisez le type de données TIMESTAMP pour stocker des valeurs d'horodatage complètes incluant la date et l'heure de la journée.

Les colonnes TIMESTAMP stockent des valeurs avec un maximum de six digits de précision pour les secondes fractionnées.

Les colonnes TIMESTAMP stockent des valeurs avec un maximum de six digits de précision pour les secondes fractionnées. Cette valeur d'horodatage complète a des valeurs par défaut (00) pour les heures, minutes et secondes manquantes. Les valeurs de fuseau horaire dans les chaînes d'entrée sont ignorées.

Par défaut, les valeurs TIMESTAMP sont UTC dans les tables utilisateur et dans les tables système Amazon Redshift.

TIMESTAMPTZ

TIMESTAMPTZ est un alias de TIMESTAMP WITH TIME ZONE.

Utilisez le type de données TIMESTAMPTZ pour saisir des valeurs d'horodatage complètes incluant la date, l'heure de la journée et un fuseau horaire. Lorsqu'une valeur d'entrée comprend un fuseau horaire, Amazon Redshift utilise le fuseau horaire pour convertir la valeur en UTC et stocke la valeur UTC.

Pour afficher la liste des noms de fuseaux horaires pris en charge, exécutez la commande suivante.

```
select pg_timezone_names();
```

Pour afficher la liste des abréviations de fuseaux horaires prises en charge, exécutez la commande suivante.

```
select pg_timezone_abbrevs();
```

Vous pouvez également trouver des informations sur les fuseaux horaires dans la base de données des fuseaux horaires IANA.

Le tableau suivant présente des exemples de formats de fuseaux horaires.

Format	exemple
jj lun hh:mi:ss aaaa tz	17 déc 07:37:16 1997 PST
mm/dd/yyyyhh:mi:ss.ss tz	12/17/1997 07:37:16.00 PST
mm/dd/yyyyhh:mi:ss.ss tz	12/17/1997 07:37:16.00 États-Unis/Pacifique

Format	exemple
yyyy-mm-dd hh : mi : ss+/-tz	1997-12-17 07:37:16-08
jj.mm.aaaa hh:mi:ss tz	17.12.1997 07:37:16.00 PST

Les colonnes TIMESTAMPTZ stockent des valeurs avec un maximum de six digits de précision pour les secondes fractionnées.

Si vous insérez une date dans une colonne TIMESTAMPTZ, ou une date avec un horodatage partiel, la valeur est implicitement convertie en une valeur d'horodatage complète. Cette valeur d'horodatage complète a des valeurs par défaut (00) pour les heures, minutes et secondes manquantes.

Les valeurs TIMESTAMPTZ sont au format UTC dans les tables utilisateur.

Exemples avec les types datetime

Vous trouverez ci-dessous des exemples d'utilisation des types datetime pris en charge par Amazon Redshift.

Exemples de date

Les exemples suivants insèrent des dates qui ont des formats différents et affichent le résultat.

```
create table datetable (start_date date, end_date date);
```

```
insert into datetable values ('2008-06-01','2008-12-31');

insert into datetable values ('Jun 1,2008','20081231');
```

```
select * from datetable order by 1;

start_date |  end_date
-----------------------
2008-06-01 | 2008-12-31
2008-06-01 | 2008-12-31
```

Si vous insérez une valeur d'horodatage dans une colonne DATE, la partie temps est ignorée et seule la date est chargée.

Exemples d'heure

Les exemples suivants insèrent des valeurs TIME et TIMETZ qui n'ont pas le même format et affichent la sortie.

```
create table timetable (start_time time, end_time timetz);
```

```
insert into timetable values ('19:11:19','20:41:19 UTC');
insert into timetable values ('191119', '204119 UTC');
```

```
select * from timetable order by 1;
start_time |  end_time
-----------------------
 19:11:19  | 20:41:19+00
 19:11:19  | 20:41:19+00
```

Exemples d'horodatages

Si vous insérez une date dans une colonne TIMESTAMP ou TIMESTAMPTZ, l'heure par défaut est minuit. Par exemple, si vous insérez le littéral `20081231`, la valeur stockée est `2008-12-31 00:00:00`.

Pour modifier le fuseau horaire de la session en cours, utilisez la commande SET pour définir le paramètre de configuration timezone.

L'exemple suivant insère des horodatages qui ont des formats différents et affiche le tableau qui en résulte.

```
create table tstamp(timeofday timestamp, timeofdaytz timestamptz);

insert into tstamp values('Jun 1,2008  09:59:59', 'Jun 1,2008 09:59:59 EST' );
insert into tstamp values('Dec 31,2008 18:20','Dec 31,2008 18:20');
insert into tstamp values('Jun 1,2008  09:59:59 EST', 'Jun 1,2008 09:59:59');

SELECT * FROM tstamp;

+---------------------+-----------------------+
|      timeofday      |      timeofdaytz      |
+---------------------+-----------------------+
| 2008-06-01 09:59:59 | 2008-06-01 14:59:59+00 |
| 2008-12-31 18:20:00 | 2008-12-31 18:20:00+00 |
```

```
| 2008-06-01 09:59:59 | 2008-06-01 09:59:59+00 |
+---------------------+------------------------+
```

Littéraux de type date, heure et horodatage

Vous trouverez ci-dessous les règles d'utilisation des littéraux de type date, heure et horodatage pris en charge par Amazon Redshift.

Dates

Les dates d'entrée suivantes sont toutes des exemples valides de valeurs de date littérales pour le type de données DATE que vous pouvez charger dans les tables Amazon Redshift. La valeur `MDY DateStyle` par défaut est supposée être en vigueur. Ce mode signifie que la valeur month précède la valeur day dans les chaînes telles que `1999-01-08` et `01/02/00`.

> **Note**
>
> Une date ou un horodatage littéral doit être placé entre guillemets lorsque vous le chargez dans une table.

Date en entrée	Date complète
8 janvier 1999	8 janvier 1999
1999-01-08	8 janvier 1999
1/8/1999	8 janvier 1999
01/02/00	2 janvier 2000
2000-Jan-31	31 janvier 2000
Jan-31-2000	31 janvier 2000
31-Jan-2000	31 janvier 2000
20080215	15 février 2008
080215	15 février 2008

Date en entrée	Date complète
2008.366	31 décembre 2008 (la partie à trois chiffres de la date doit être comprise entre 001 et 366)

Times

Les temps de saisie suivants sont tous des exemples valides de valeurs temporelles littérales pour les types de données TIME et TIMETZ que vous pouvez charger dans les tables Amazon Redshift.

Heures en entrée	Description (de la partie heure)
04:05:06.789	4:05 AM et 6,789 secondes
04:05:06	4:05 AM et 6 secondes
04:05	4:05 AM exactement
040506	4:05 AM et 6 secondes
04:05 AM	4:05 AM exactement ; AM est facultatif
04:05 PM	4:05 PM exactement ; la valeur d’heure doit être inférieure à 12
16h05	4:05 PM exactement

Horodatages

Les horodatages d'entrée suivants sont tous des exemples valides de valeurs temporelles littérales pour les types de données TIMESTAMP et TIMESTAMPTZ que vous pouvez charger dans les tables Amazon Redshift. Tous les littéraux de date valides peuvent être combinés avec les littéraux d’heure suivants.

Horodatages en entrée (dates et heures concaténées)	Description (de la partie heure)
20080215 04:05:06.789	4:05 AM et 6,789 secondes

Horodatages en entrée (dates et heures concaténées)	Description (de la partie heure)
20080215 04:05:06	4:05 AM et 6 secondes
20080215 04:05	4:05 AM exactement
20080215 040506	4:05 AM et 6 secondes
20080215 04:05 AM	4:05 AM exactement ; AM est facultatif
20080215 04:05 PM	4:05 PM exactement ; la valeur d'heure doit être inférieure à 12
20080215 16:05	4:05 PM exactement
20080215	Minuit (par défaut)

Valeurs datetime spéciales

Les valeurs spéciales suivantes peuvent être utilisées comme littéraux datetime et comme arguments des fonctions date. Elles requièrent des apostrophes droites et sont converties en valeurs timestamp régulières lors du traitement de la requête.

Valeur spéciale	Description
`now`	Correspond à l'heure de début de la transacti on actuelle et retourne un horodatage avec une précision de l'ordre de la microseconde.
`today`	Correspond à la date appropriée et renvoie un horodatage avec des zéros pour la partie heure.
`tomorrow`	Correspond à la date appropriée et renvoie un horodatage avec des zéros pour la partie heure.

Valeur spéciale	Description
`yesterday`	Correspond à la date appropriée et renvoie un horodatage avec des zéros pour la partie heure.

Les exemples suivants illustrent comment `now` et `today` fonctionnent avec la fonction DATEADD.

```
select dateadd(day,1,'today');

date_add
---------------------
2009-11-17 00:00:00
(1 row)

select dateadd(day,1,'now');

date_add
----------------------------
2009-11-17 10:45:32.021394
(1 row)
```

Types de données d'intervalle et littéraux

Vous pouvez utiliser un type de données d'intervalle pour stocker les durées dans des unités telles que,`seconds`,`minutes`,`hours`, `daysmonths`, et`years`. Les types de données et les littéraux d'intervalle peuvent être utilisés dans les calculs de date/heure, tels que l'ajout d'intervalles aux dates et aux horodatages, la somme des intervalles et la soustraction d'un intervalle d'une date ou d'un horodatage. Les littéraux d'intervalle peuvent être utilisés comme valeurs d'entrée pour intercaler les colonnes de type de données d'une table.

Syntaxe du type de données d'intervalle

Pour spécifier un type de données d'intervalle afin de stocker une durée en années et en mois :

```
INTERVAL year_to_month_qualifier
```

Pour spécifier un type de données d'intervalle afin de stocker une durée en jours, heures, minutes et secondes :

```
INTERVAL day_to_second_qualifier [ (fractional_precision) ]
```

Syntaxe du littéral d'intervalle

Pour spécifier un intervalle littéral afin de définir une durée en années et en mois :

```
INTERVAL quoted-string year_to_month_qualifier
```

Pour spécifier un intervalle littéral afin de définir une durée en jours, heures, minutes et secondes :

```
INTERVAL quoted-string day_to_second_qualifier [ (fractional_precision) ]
```

Arguments

chaîne entre guillemets

Spécifie une valeur numérique positive ou négative spécifiant une quantité et l'unité date/heure en tant que chaîne d'entrée. Si la chaîne entre guillemets ne contient qu'un chiffre, Amazon Redshift détermine les unités à partir du year_to_month_qualifier ou du day_to_second_qualifier. Par exemple, `'23' MONTH` représente`1 year 11 months`, `'-2' DAY` représente `-2 days 0 hours 0 minutes 0.0 seconds`1 `year 2 months`, `'1-2' MONTH` représente et `'13 day 1 hour 1 minute 1.123 seconds' SECOND` représente`13 days 1 hour 1 minute 1.123 seconds`. Pour plus d'informations sur les formats de sortie d'un intervalle, consultezStyles d'intervalle.

qualificatif annuel au mois

Spécifie la plage de l'intervalle. Si vous utilisez un qualificatif et créez un intervalle dont les unités de temps sont inférieures au qualificatif, Amazon Redshift tronque et supprime les plus petites parties de l'intervalle. Les valeurs valides pour year_to_month_qualifier sont les suivantes :

- `YEAR`
- `MONTH`
- `YEAR TO MONTH`

qualificatif du jour au deuxième

Spécifie la plage de l'intervalle. Si vous utilisez un qualificatif et créez un intervalle dont les unités de temps sont inférieures au qualificatif, Amazon Redshift tronque et supprime les plus petites parties de l'intervalle. Les valeurs valides pour day_to_second_qualifier sont les suivantes :

- DAY
- HOUR
- MINUTE
- SECOND
- DAY TO HOUR
- DAY TO MINUTE
- DAY TO SECOND
- HOUR TO MINUTE
- HOUR TO SECOND
- MINUTE TO SECOND

La sortie du littéral INTERVAL est tronquée au plus petit composant INTERVAL spécifié. Par exemple, lorsque vous utilisez un qualificatif MINUTE, Amazon Redshift supprime les unités de temps inférieures à MINUTE.

```
select INTERVAL '1 day 1 hour 1 minute 1.123 seconds' MINUTE
```

La valeur résultante est tronquée à. `'1 day 01:01:00'`

précision_fractionnaire

Paramètre facultatif qui spécifie le nombre de chiffres fractionnaires autorisés dans l'intervalle. L'argument fractional_precision ne doit être spécifié que si votre intervalle contient SECOND. Par exemple, `SECOND(3)` crée un intervalle qui n'autorise que trois chiffres fractionnaires, tels que 1,234 seconde. Le nombre maximum de chiffres fractionnaires est de six.

La configuration de session `interval_forbid_composite_literals` détermine si une erreur est renvoyée lorsqu'un intervalle est spécifié avec les parties YEAR TO MONTH et DAY TO SECOND. Pour de plus amples informations, veuillez consulter interval_forbid_composite_literals.

Arithmétique des intervalles

Vous pouvez utiliser des valeurs d'intervalle avec d'autres valeurs de date/heure pour effectuer des opérations arithmétiques. Les tableaux suivants décrivent les opérations disponibles et le type de données résultant de chaque opération.

> **Note**
>
> Les opérations qui peuvent produire les deux `date` et les `timestamp` résultats le font en fonction de la plus petite unité de temps impliquée dans l'équation. Par exemple, lorsque vous ajoutez un `interval` à un, `date` le résultat est un intervalle d'une `date` ANNÉE À UN MOIS, et un horodatage s'il s'agit d'un intervalle D'UN JOUR À LA SECONDE.

Les opérations où le premier opérande est an `interval` produisent les résultats suivants pour le second opérande donné :

Opérateur	Date	Horodatage	Intervalle	Numérique
-	N/A	N/A	Intervalle	N/A
+	Date	Date/Horodatage	Intervalle	N/A
*	N/A	N/A	N/A	Intervalle
/	N/A	N/A	N/A	Intervalle

Les opérations où le premier opérande est a `date` produisent les résultats suivants pour le second opérande donné :

Opérateur	Date	Horodatage	Intervalle	Numérique
-	Numérique	Intervalle	Date/Horodatage	Date
+	N/A	N/A	N/A	N/A

Les opérations où le premier opérande est a `timestamp` produisent les résultats suivants pour le second opérande donné :

Opérateur	Date	Horodatage	Intervalle	Numérique
-	Numérique	Intervalle	Horodatage	Horodatage

Opérateur	Date	Horodatage	Intervalle	Numérique
+	N/A	N/A	N/A	N/A

Styles d'intervalle

Vous pouvez utiliser la the section called “SET” commande SQL pour modifier le format d'affichage de sortie de vos valeurs d'intervalle. Lorsque vous utilisez le type de données d'intervalle dans SQL, convertissez-le en texte pour voir le style d'intervalle attendu, par exemple,`YEAR TO MONTH::text`. Les valeurs disponibles pour DÉFINIR la `IntervalStyle` valeur sont les suivantes :

- `postgres`— suit le style PostgreSQL. Il s’agit de l’option par défaut.
- `postgres_verbose`— suit le style détaillé de PostgreSQL.
- `sql_standard`— suit le style des littéraux d'intervalle standard SQL.

La commande suivante définit le style d'intervalle sur`sql_standard`.

```
SET IntervalStyle to 'sql_standard';
```

format de sortie postgres

Le format de sortie pour le style d'`postgres`intervalle est le suivant. Chaque valeur numérique peut être négative.

```
'<numeric> <unit> [, <numeric> <unit> ...]'
```

```
select INTERVAL '1-2' YEAR TO MONTH::text

varchar
---------------
1 year 2 mons
```

```
select INTERVAL '1 2:3:4.5678' DAY TO SECOND::text

varchar
------------------
1 day 02:03:04.5678
```

format de sortie postgres_verbose

La syntaxe de postgres_verbose est similaire à celle de postgres, mais les sorties postgres_verbose contiennent également l'unité de temps.

```
'[@] <numeric> <unit> [, <numeric> <unit> ...] [direction]'
```

```
select INTERVAL '1-2' YEAR TO MONTH::text

varchar
-----------------
@ 1 year 2 mons
```

```
select INTERVAL '1 2:3:4.5678' DAY TO SECOND::text

varchar
--------------------------
@ 1 day 2 hours 3 mins 4.56 secs
```

format de sortie sql_standard

Les valeurs d'intervalle d'une année à l'autre sont formatées comme suit. La spécification d'un signe négatif avant l'intervalle indique que l'intervalle est une valeur négative et s'applique à l'ensemble de l'intervalle.

```
'[-]yy-mm'
```

Les valeurs de l'intervalle entre le jour et la seconde sont formatées comme suit.

```
'[-]dd hh:mm:ss.ffffff'
```

```
SELECT INTERVAL '1-2' YEAR TO MONTH::text

varchar
-------
1-2
```

```
select INTERVAL '1 2:3:4.5678' DAY TO SECOND::text
```

```
varchar
---------------
1 2:03:04.5678
```

Exemples de type de données d'intervalle

Les exemples suivants montrent comment utiliser les types de données INTERVAL avec des tables.

```
create table sample_intervals (y2m interval month, h2m interval hour to minute);
insert into sample_intervals values (interval '20' month, interval '2 days
 1:1:1.123456' day to second);
select y2m::text, h2m::text from sample_intervals;


      y2m      |       h2m
---------------+-----------------
 1 year 8 mons | 2 days 01:01:00
```

```
update sample_intervals set y2m = interval '2' year where y2m = interval '1-8' year to
 month;
select * from sample_intervals;

   y2m   |       h2m
---------+-----------------
 2 years | 2 days 01:01:00
```

```
delete from sample_intervals where h2m = interval '2 1:1:0' day to second;
select * from sample_intervals;

 y2m | h2m
-----+-----
```

Exemples de littéraux d'intervalle

Les exemples suivants sont exécutés avec le style d'intervalle défini sur`postgres`.

L'exemple suivant montre comment créer un littéral INTERVAL de 1 an.

```
select INTERVAL '1' YEAR
```

```
intervaly2m
---------------
1 years 0 mons
```

Si vous spécifiez une chaîne entre guillemets qui dépasse le qualificatif, les unités de temps restantes sont tronquées par rapport à l'intervalle. Dans l'exemple suivant, un intervalle de 13 mois devient 1 an et 1 mois, mais le mois restant est omis en raison du qualificatif YEAR.

```
select INTERVAL '13 months' YEAR

intervaly2m
---------------
1 years 0 mons
```

Si vous utilisez un qualificatif inférieur à votre chaîne d'intervalle, les unités restantes sont incluses.

```
select INTERVAL '13 months' MONTH

intervaly2m
---------------
1 years 1 mons
```

Si vous spécifiez une précision dans votre intervalle, le nombre de chiffres fractionnaires est tronqué à la précision spécifiée.

```
select INTERVAL '1.234567' SECOND (3)

intervald2s
--------------------------------
0 days 0 hours 0 mins 1.235 secs
```

Si vous ne spécifiez aucune précision, Amazon Redshift utilise la précision maximale de 6.

```
select INTERVAL '1.23456789' SECOND

intervald2s
-----------------------------------
0 days 0 hours 0 mins 1.234567 secs
```

L'exemple suivant montre comment créer un intervalle échelonné.

```
select INTERVAL '2:2' MINUTE TO SECOND

intervald2s
-----------------------------
0 days 0 hours 2 mins 2.0 secs
```

Les qualificatifs dictent les unités que vous spécifiez. Par exemple, même si l'exemple suivant utilise la même chaîne entre guillemets de « 2:2 » que l'exemple précédent, Amazon Redshift reconnaît qu'il utilise des unités de temps différentes en raison du qualificatif.

```
select INTERVAL '2:2' HOUR TO MINUTE

intervald2s
-----------------------------
0 days 2 hours 2 mins 0.0 secs
```

Les abréviations et les pluriels de chaque unité sont également pris en charge. Par exemple, `5s` `5 second`, et `5 seconds` sont des intervalles équivalents. Les unités prises en charge sont les années, les mois, les heures, les minutes et les secondes.

```
select INTERVAL '5s' SECOND

intervald2s
-----------------------------
0 days 0 hours 0 mins 5.0 secs
```

```
select INTERVAL '5 HOURS' HOUR

intervald2s
-----------------------------
0 days 5 hours 0 mins 0.0 secs
```

```
select INTERVAL '5 h' HOUR

intervald2s
-----------------------------
0 days 5 hours 0 mins 0.0 secs
```

Exemples de littéraux d'intervalle sans syntaxe de qualificatif

> **Note**
>
> Les exemples suivants illustrent l'utilisation d'un intervalle littéral sans `DAY TO SECOND` qualificatif `YEAR TO MONTH` ou. Pour plus d'informations sur l'utilisation du littéral d'intervalle recommandé avec un qualificatif, consultezTypes de données d'intervalle et littéraux.

Utilisez un littéral de type interval pour identifier les périodes spécifiques, comme `12 hours` ou `6 months`. Vous pouvez utiliser ces littéraux de type interval dans les cas et les calculs qui impliquent des expressions de type datetime.

Un littéral d'intervalle est exprimé sous la forme d'une combinaison du mot clé INTERVAL avec une quantité numérique et une partie de date prise en charge, par exemple `INTERVAL '7 days'` ou. `INTERVAL '59 minutes'` Plusieurs quantités et unités peuvent être associées pour former un intervalle plus précis ; par exemple : `INTERVAL '7 days, 3 hours, 59 minutes'`. Les abréviations et les pluriels de chaque unité sont également pris en charge ; par exemple : `5 s`, `5 second` et `5 seconds` sont des intervalles équivalents.

Si vous ne spécifiez pas une partie date, la valeur de l'intervalle correspond à des secondes. Vous pouvez spécifier la valeur de la quantité sous forme de fraction (par exemple : `0.5 days`).

Les exemples suivants illustrent une série de calculs avec différentes valeurs d'intervalle.

Ce qui suit ajoute 1 seconde à la date spécifiée.

```
select caldate + interval '1 second' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 00:00:01
(1 row)
```

Ce qui suit ajoute 1 minute à la date spécifiée.

```
select caldate + interval '1 minute' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
```

```
2008-12-31 00:01:00
(1 row)
```

Ce qui suit ajoute 3 heures et 35 minutes à la date spécifiée.

```
select caldate + interval '3 hours, 35 minutes' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 03:35:00
(1 row)
```

Ce qui suit ajoute 52 semaines à la date spécifiée.

```
select caldate + interval '52 weeks' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2009-12-30 00:00:00
(1 row)
```

Ce qui suit ajoute 1 semaine, 1 heure, 1 minute, et 1 seconde à la date spécifiée.

```
select caldate + interval '1w, 1h, 1m, 1s' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2009-01-07 01:01:01
(1 row)
```

Ce qui suit ajoute 12 heures (la moitié d'une journée) à la date spécifiée.

```
select caldate + interval '0.5 days' as dateplus from date
where caldate='12-31-2008';
dateplus
---------------------
2008-12-31 12:00:00
(1 row)
```

Ce qui suit soustrait 4 mois à compter du 15 février 2023 et le résultat est le 15 octobre 2022.

```
select date '2023-02-15' - interval '4 months';

?column?
---------------------
2022-10-15 00:00:00
```

Ce qui suit soustrait 4 mois à compter du 31 mars 2023 et le résultat est le 30 novembre 2022. Le calcul prend en compte le nombre de jours dans un mois.

```
select date '2023-03-31' - interval '4 months';

?column?
---------------------
2022-11-30 00:00:00
```

Type Boolean

Utilisez le type de données BOOLEAN pour stocker les valeurs true et false dans une colonne codée sur un octet. Le tableau suivant décrit les trois états possibles pour une valeur booléenne et les valeurs littérales qui entraînent cet état. Quelle que soit la chaîne en entrée, une colonne booléenne stocke et émet « t » pour true et « f » pour false.

État	Valeurs littérales valides	Stockage
True	`TRUE 't' 'true' 'y' 'yes' '1'`	1 octet
False	`FALSE 'f' 'false' 'n' 'no' '0'`	1 octet
Je ne sais pas	`NULL`	1 octet

Vous pouvez utiliser une comparaison IS pour vérifier une valeur booléenne uniquement sous la forme d’un prédicat dans la clause WHERE. Vous ne pouvez pas utiliser la comparaison IS avec une valeur booléenne dans la liste SELECT.

Exemples

Vous pouvez utiliser une colonne de type BOOLEAN pour stocker un état « Actif/Inactif » pour chaque client dans une table CUSTOMER.

```
create table customer(
custid int,
active_flag boolean default true);
```

```
insert into customer values(100, default);
```

```
select * from customer;
custid | active_flag
-------+-------------
   100 | t
```

Si aucune valeur par défaut (`true` ou `false`) n'est spécifiée dans l'instruction CREATE TABLE, l'insertion d'une valeur par défaut signifie l'insertion d'une valeur null.

Dans cet exemple, la requête sélectionne les utilisateurs de la table USERS qui aiment le sport, mais n'aiment pas le théâtre :

```
select firstname, lastname, likesports, liketheatre
from users
where likesports is true and liketheatre is false
order by userid limit 10;

firstname |  lastname  | likesports | liketheatre
----------+------------+------------+-------------
Lars      | Ratliff    | t          | f
Mufutau   | Watkins    | t          | f
Scarlett  | Mayer      | t          | f
Shafira   | Glenn      | t          | f
Winifred  | Cherry     | t          | f
Chase     | Lamb       | t          | f
Liberty   | Ellison    | t          | f
Aladdin   | Haney      | t          | f
Tashya    | Michael    | t          | f
Lucian    | Montgomery | t          | f
(10 rows)
```

L'exemple suivant sélectionne les utilisateurs de la table USERS pour lesquels on ignore s'ils aiment la musique rock.

```
select firstname, lastname, likerock
from users
where likerock is unknown
order by userid limit 10;

firstname | lastname | likerock
----------+----------+----------
Rafael    | Taylor   |
Vladimir  | Humphrey |
Barry     | Roy      |
Tamekah   | Juarez   |
Mufutau   | Watkins  |
Naida     | Calderon |
Anika     | Huff     |
Bruce     | Beck     |
Mallory   | Farrell  |
Scarlett  | Mayer    |
(10 rows)
```

L'exemple suivant renvoie une erreur parce qu'il utilise une comparaison IS dans la liste SELECT.

```
select firstname, lastname, likerock is true as "check"
from users
order by userid limit 10;

[Amazon](500310) Invalid operation: Not implemented
```

L'exemple suivant réussit parce qu'il utilise une comparaison d'égalité (=) dans la liste SELECT à la place de la comparaison IS.

```
select firstname, lastname, likerock = true as "check"
from users
order by userid limit 10;

firstname | lastname  | check
----------+-----------+------
Rafael    | Taylor    |
Vladimir  | Humphrey  |
Lars      | Ratliff   | true
```

```
Barry     | Roy       |
Reagan    | Hodge     | true
Victor    | Hernandez | true
Tamekah   | Juarez    |
Colton    | Roy       | false
Mufutau   | Watkins   |
Naida     | Calderon  |
```

Type HLLSKETCH

Utilisez le type de données HLLSKETCH pour HyperLogLog les esquisses. Amazon Redshift prend en charge les représentations d' HyperLogLog esquisse éparses ou denses. Les esquisses sont d'abord fragmentées, puis deviennent denses lorsque le format dense est plus efficace pour minimiser l'empreinte mémoire utilisée.

Amazon Redshift effectue automatiquement la transition d'une HyperLogLog esquisse fragmentée lors de l'importation, de l'exportation ou de l'impression d'esquisses au format JSON suivant.

```
{"logm":15,"sparse":{"indices":[4878,9559,14523],"values":[1,2,1]}}
```

Amazon Redshift utilise une représentation sous forme de chaîne au format Base64 pour représenter une esquisse dense. HyperLogLog

Amazon Redshift utilise la représentation sous forme de chaîne suivante au format Base64 pour représenter une esquisse dense. HyperLogLog

```
"ABAABA..."
```

La taille maximale d'un objet HLLSKETCH est de 24 580 octets lorsqu'il est utilisé dans la compression brute.

Type SUPER

Utilisez le type de données SUPER pour stocker des données semi-structurées ou des documents en tant que valeurs.

Les données semi-structurées ne sont pas conformes à la structure rigide et tabulaire du modèle de données relationnelles utilisé dans les bases de données SQL. Elles contiennent des balises qui font référence à des entités distinctes dans les données. Elles peuvent contenir des valeurs complexes telles que des tableaux, des structures imbriquées et d'autres structures complexes associées à des

formats de sérialisation, tels que JSON. Le type de données SUPER est un ensemble de valeurs de tableau et de structure sans schéma qui englobent tous les autres types scalaires d'Amazon Redshift.

Le type de données SUPER prend en charge jusqu'à 16 Mo de données pour un objet SUPER individuel. Pour plus d'informations sur le type de données SUPER, y compris des exemples d'implémentation dans une table, consultez Données semi-structurées dans Amazon Redshift.

Les objets SUPER de plus de 1 Mo ne peuvent être ingérés qu'à partir des formats de fichier suivants :

- Parquet
- JSON
- TEXT
- CSV

Le type de données SUPER a les propriétés suivantes :

- Une valeur scalaire Amazon Redshift :
  - Un null
  - Une valeur booléenne
  - Un nombre, par exemple, smallint, entier, bigint, décimal ou virgule flottante (tel que float4 ou float8)
  - Une valeur de chaîne, telle que varchar ou char
- Une valeur complexe :
  - Un tableau de valeurs, y compris scalaire ou complexe
  - Structure, également appelée tuple ou objet, qui est une carte de noms et de valeurs d'attribut (scalaire ou complexe)

Chacun des deux types de valeurs complexes contient ses propres scalaires ou valeurs complexes sans aucune restriction de régularité.

Le type de données SUPER prend en charge la persistance des données semi-structurées sous une forme sans schéma. Bien que le modèle de données hiérarchique puisse changer, les anciennes versions de données peuvent coexister dans la même colonne SUPER.

Amazon Redshift utilise PartiQL pour activer la navigation dans les tableaux et les structures. Amazon Redshift utilise aussi la syntaxe PartiQL pour itérer dans les tableaux SUPER. Pour plus d'informations, consultez Navigation et Désimbriquer des requêtes.

Amazon Redshift utilise le typage dynamique pour traiter les données SUPER sans schéma sans avoir à déclarer les types de données avant qu'elles ne soient utilisées dans votre requête. Pour de plus amples informations, veuillez consulter Typage dynamique.

Vous pouvez appliquer des politiques de masquage dynamique des données aux valeurs `scalar` figurant sur les chemins des colonnes de type SUPER. Pour plus d'informations sur le masquage dynamique des données, consultez Masquage dynamique des données. Pour en savoir plus sur l'utilisation du masquage dynamique des données avec le type de données SUPER, consultez Utilisation du masquage dynamique des données avec des chemins de type de données SUPER.

Type VARBYTE

Utilisez une colonne VARBYTE, VARBINARY ou BINARY VARYING pour stocker une valeur binaire de longueur variable avec une limite fixe.

```
varbyte [ (n) ]
```

Le nombre maximum d'octets (n) peut être compris entre 1 et 16 777 216. La valeur par défaut est 64 000.

Voici quelques exemples dans lesquels vous souhaiterez peut-être utiliser un type de données VARBYTE :

- Joindre des tables sur des colonnes VARBYTE.
- Création de vues matérialisées contenant des colonnes VARBYTE. L'actualisation progressive des vues matérialisées contenant des colonnes VARBYTE est prise en charge. Toutefois, les fonctions d'agrégation autres que COUNT, MIN et MAX et GROUP BY sur les colonnes VARBYTE ne prennent pas en charge l'actualisation progressive.

Pour garantir que tous les octets sont des caractères imprimables, Amazon Redshift utilise le format hexadécimal pour imprimer des valeurs VARBYTE. Par exemple, le code SQL suivant convertit la chaîne hexadécimale 6162 en une valeur binaire. Même si la valeur renvoyée est une valeur binaire, les résultats sont imprimés en hexadécimal 6162.

```
select from_hex('6162');
```

```
 from_hex
----------
 6162
```

Amazon Redshift prend en charge la conversion entre le type de données VARBYTE et les types de données suivants :

- CHAR
- VARCHAR
- SMALLINT
- INTEGER
- BIGINT

Lors de la conversion CHAR et VARCHAR, le format UTF-8 est utilisé. Pour plus d'informations sur le format UTF-8, consultez TO_VARBYTE. Lors de la conversion à partir de SMALLINT, INTEGER et BIGINT, le nombre d'octets du type de données d'origine est conservé. Il s'agit de deux octets pour SMALLINT, de quatre octets pour INTEGER et de huit octets pour BIGINT.

L'instruction SQL suivante convertit une chaîne VARCHAR en VARBYTE. Même si la valeur renvoyée est une valeur binaire, les résultats sont imprimés en hexadécimal 616263.

```
select 'abc'::varbyte;

 varbyte
---------
 616263
```

L'instruction SQL suivante convertit une valeur CHAR dans une colonne en VARBYTE. Cet exemple montre comment créer une table avec une colonne (c) CHAR(10) et insérer des valeurs de caractères inférieures à 10. La conversion résultante bloque le résultat avec des caractères d'espace (hex'20') à la taille de colonne définie. Même si la valeur renvoyée est une valeur binaire, les résultats sont imprimés en hexadécimal.

```
create table t (c char(10));
insert into t values ('aa'), ('abc');
select c::varbyte from t;
          c
```

```
----------------------
 61612020202020202020
 61626320202020202020
```

L'instruction SQL suivante convertit une chaîne SMALLINT en VARBYTE. Même si la valeur renvoyée est une valeur binaire, les résultats sont imprimés en hexadécimal `0005`, soit deux octets ou quatre caractères hexadécimaux.

```
select 5::smallint::varbyte;

 varbyte
---------
 0005
```

L'instruction SQL suivante convertit un INTEGER en VARBYTE. Même si la valeur renvoyée est une valeur binaire, les résultats sont imprimés en hexadécimal `00000005`, soit quatre octets ou huit caractères hexadécimaux.

```
select 5::int::varbyte;

 varbyte
----------
 00000005
```

L'instruction SQL suivante convertit un BIGINT en VARBYTE. Même si la valeur renvoyée est une valeur binaire, les résultats sont imprimés en hexadécimal `0000000000000005`, soit 8 octets ou 16 caractères hexadécimaux.

```
select 5::bigint::varbyte;

     varbyte
------------------
 0000000000000005
```

Les fonctions Amazon Redshift prenant en charge le type de données VARBYTE sont les suivantes :

- Opérateurs VARBYTE
- CONCAT

- LEN
- Fonction LENGTH
- OCTET_LENGTH
- Fonction SUBSTRING
- FROM_HEX
- TO_HEX
- FROM_VARBYTE
- TO_VARBYTE
- GETBIT
- Chargement d’une colonne avec le type de données VARBYTE
- Déchargement d’une colonne avec le type de données VARBYTE

Limites lors de l’utilisation du type de données VARBYTE avec Amazon Redshift

Les limites suivantes se présentent lors de l’utilisation du type de données VARBYTE avec Amazon Redshift :

- Amazon Redshift Spectrum prend en charge le type de données VARBYTE uniquement pour les fichiers Parquet et ORC.
- L’éditeur de requêtes Amazon Redshift et l’éditeur de requêtes Amazon Redshift v2 ne prennent pas encore entièrement en charge le type de données VARBYTE. Par conséquent, utilisez un client SQL différent lorsque vous utilisez des expressions VARBYTE.

  Pour contourner l’utilisation de l’éditeur de requêtes, si la longueur de vos données est inférieure à 64 Ko et que le contenu est en UTF-8 valide, vous pouvez convertir les valeurs VARBYTE en VARCHAR, par exemple :

  ```
  select to_varbyte('6162', 'hex')::varchar;
  ```
- Vous ne pouvez pas utiliser les types de données VARBYTE avec des fonctions définies par l'utilisateur Python ou Lambda (). UDFs
- Vous ne pouvez pas créer de colonne HLLSKETCH à partir d’une colonne VARBYTE ou utiliser APPROXIMATIVE COUNT DISTINCT sur une colonne VARBYTE.
- Les valeurs VARBYTE supérieures à 1 Mo ne peuvent être ingérées qu'à partir des formats de fichier suivants :

- Parquet
- Texte
- Valeurs séparées par des virgules (CSV)

Compatibilité et conversion de types

Vous trouverez ci-dessous une discussion sur la façon dont les règles de conversion de type et la compatibilité des types de données fonctionnent dans Amazon Redshift.

Compatibilité

La correspondance des types de données et la correspondance des valeurs littérales et des constantes avec les types de données se produisent lors de différentes opérations de base de données, dont les suivantes :

- Opérations DML (Data Manipulation Language) sur les tables
- Requêtes UNION, INTERSECT et EXCEPT
- Expressions CASE
- Evaluation de prédicats, tels que LIKE et IN
- Evaluation de fonctions SQL qui effectuent des comparaisons ou des extractions de données
- Comparaisons avec les opérateurs mathématiques

Les résultats de ces opérations dépendent des règles de conversion de types et de la compatibilité des types de données. La compatibilité implique que la mise en one-to-one correspondance d'une certaine valeur et d'un certain type de données n'est pas toujours requise. Comme certains types de données sont compatibles, une conversion implicite ou forçage de type, est possible (pour plus d’informations, consultez Types de conversion implicite). Lorsque les types de données sont incompatibles, vous pouvez parfois convertir une valeur d’un type de données en un autre à l’aide d’une fonction de conversion explicite.

Compatibilité générale et règles de conversion

Notez les règles de compatibilité et de conversion suivantes :

- En général, les types de données qui appartiennent à la même catégorie (comme les différents types de données numériques) sont compatibles et peuvent être convertis implicitement.

Par exemple, avec une conversion implicite, vous pouvez insérer une valeur décimale dans une colonne de type entier. La partie décimale est arrondie pour produire un nombre entier. Ou vous pouvez extraire une valeur numérique, telle que `2008`, d'une date et insérer cette valeur dans une colonne de type entier.

- Les types de données numériques renforcent les conditions de débordement qui se produisent lorsque vous tentez d'insérer out-of-range des valeurs. Par exemple, une valeur décimale avec une précision de 5 ne peut contenir dans une colonne décimale dont la précision est 4. Un nombre entier ou la partie entière d'un nombre décimal ne sont jamais tronqués ; toutefois, la partie fractionnaire d'un nombre décimal peut être arrondie vers le haut ou vers le bas, selon le cas. Cependant, les résultats de conversions explicites de valeurs sélectionnées dans les tables ne sont pas arrondis.
- Différents types de chaînes de caractères sont compatibles ; les chaînes de colonnes VARCHAR contenant des données codées sur un seul octet et les chaînes de colonne CHAR sont comparables et implicitement convertibles. Les chaînes VARCHAR qui contiennent des données codées sur plusieurs octets ne sont pas comparables. Vous pouvez également convertir une chaîne de caractères en une date, une heure, un horodatage ou une valeur numérique si la chaîne est une valeur littérale appropriée ; les espaces de début ou de fin sont ignorés. Inversement, vous pouvez convertir une date, une heure, un horodatage ou une valeur numérique en une chaîne de caractères de longueur fixe ou variable.

  > **Note**
  >
  > Une chaîne de caractères que vous voulez convertir en type numérique doit comporter la représentation en caractères d'un nombre. Par exemple, vous pouvez convertir les chaînes `'1.0'` ou `'5.9'` en valeurs décimales, mais vous ne pouvez pas convertir la chaîne `'ABC'` en un type numérique.

- Si vous comparez des valeurs DECIMAL avec des chaînes de caractères, Amazon Redshift tente de convertir la chaîne de caractères en valeur DECIMAL. Lors de la comparaison de toutes les autres valeurs numériques avec des chaînes de caractères, les valeurs numériques sont converties en chaînes de caractères. Pour effectuer la conversion inverse (par exemple, convertir des chaînes de caractères en entiers ou convertir des valeurs DECIMALES en chaînes de caractères), utilisez une fonction explicite, telle que CAST.
- Pour convertir les valeurs DECIMAL ou NUMERIC 64 bits en une plus grande précision, vous devez utiliser une fonction de conversion explicite telle que les fonctions CAST ou CONVERT.

- Lors de la conversion de DATE ou TIMESTAMP en TIMESTAMPTZ, ou de la conversion de TIME en TIMESTAMP, le fuseau horaire de la session en cours. Le fuseau horaire de session est UTC par défaut. Pour plus d'informations sur le réglage du fuseau horaire de la session, consultez timezone.
- De même, TIMESTAMPTZ est converti en DATE, TIME ou TIMESTAMP en fonction du fuseau horaire de la session en cours. Le fuseau horaire de session est UTC par défaut. Après la conversion, les informations sur les fuseaux horaires sont abandonnées.
- Les chaînes de caractères qui représentent un horodatage avec un fuseau horaire spécifié sont converties en TIMESTAMPTZ en utilisant le fuseau horaire de la session actuelle, qui est UTC par défaut. De même, les chaînes de caractères qui représentent un fuseau horaire spécifié sont converties en TIPZ à l'aide du fuseau horaire de la session en cours, UTC par défaut.

Types de conversion implicite

Il existe deux types de conversion implicite :

- Conversions implicites d'affectations, telles que la définition de valeurs dans les commandes INSERT ou UPDATE.
- Conversions implicites d'expressions, comme l'exécution de comparaisons dans la clause WHERE.

Le tableau suivant répertorie les types de données qui peuvent être convertis implicitement dans les affectations ou les expressions. Vous pouvez également utiliser une fonction de conversion explicite pour exécuter ces conversions.

Type de départ	Type d'arrivée
BIGINT (1) INT8	BOOLEAN
	CHAR
	DECIMAL (NUMERIC)
	DOUBLE PRÉCISION (FLOAT8)
	ENTIER (INT, INT4)
	RÉEL (FLOAT4)

Type de départ	Type d'arrivée
	PETITE MENTHE () INT2
	VARCHAR
CHAR	VARCHAR
DATE	CHAR
	VARCHAR
	TIMESTAMP
	TIMESTAMPTZ
DECIMAL (NUMERIC)	BIGINT (1) INT8
	CHAR
	DOUBLE PRÉCISION (FLOAT8)
	ENTIER (INT, INT4)
	RÉEL (FLOAT4)
	PETITE MENTHE () INT2
	VARCHAR
DOUBLE PRÉCISION (FLOAT8)	BIGINT (1) INT8
	CHAR
	DECIMAL (NUMERIC)
	ENTIER (INT, INT4)
	RÉEL (FLOAT4)
	PETITE MENTHE () INT2

Type de départ	Type d’arrivée
	VARCHAR
ENTIER (INT, INT4)	BIGINT (1) INT8
	BOOLEAN
	CHAR
	DECIMAL (NUMERIC)
	DOUBLE PRÉCISION (FLOAT8)
	RÉEL (FLOAT4)
	PETITE MENTHE () INT2
	VARCHAR
RÉEL (FLOAT4)	BIGINT (1) INT8
	CHAR
	DECIMAL (NUMERIC)
	ENTIER (INT, INT4)
	PETITE MENTHE () INT2
	VARCHAR
PETITE MENTHE () INT2	BIGINT (1) INT8
	BOOLEAN
	CHAR
	DECIMAL (NUMERIC)
	DOUBLE PRÉCISION (FLOAT8)

Type de départ	Type d’arrivée
	ENTIER (INT, INT4)
	RÉEL (FLOAT4)
	VARCHAR
TIMESTAMP	CHAR
	DATE
	VARCHAR
	TIMESTAMPTZ
	TIME
TIMESTAMPTZ	CHAR
	DATE
	VARCHAR
	TIMESTAMP
	TIMETZ
TIME	VARCHAR
	TIMETZ
	INTERVALLE D'UN JOUR À L'AUTRE
TIMETZ	VARCHAR
	TIME
GEOMETRY	GEOGRAPHY
GEOGRAPHY	GEOMETRY

> **Note**
>
> Les conversions implicites entre TIMESTAMPTZ, TIMESTAMP, DATE, TIME, TIMETZ ou les chaînes de caractères utilisent le fuseau horaire actuel de la session. Pour plus d'informations sur la définition du fuseau horaire en cours, consultez timezone.
> Les types de données GEOMETRY et GEOGRAPHY ne peuvent pas être convertis de façon implicite dans un autre type de données, excepté l'un dans l'autre. Pour de plus amples informations, veuillez consulter Fonction CAST.
> Le type de données VARBYTE ne peut pas être converti de façon implicite dans un autre type de données. Pour de plus amples informations, veuillez consulter Fonction CAST.

Utilisation du typage dynamique pour le type de données SUPER

Amazon Redshift utilise le typage dynamique pour traiter des données SUPER sans schéma sans avoir à déclarer les types de données avant de les utiliser dans votre requête. Le typage dynamique utilise les résultats de la navigation dans les colonnes de données SUPER sans devoir les convertir explicitement en types Amazon Redshift. Pour plus d'informations sur l'utilisation du typage dynamique pour le type de données SUPER, consultez Typage dynamique.

Vous pouvez lancer des valeurs SUPER depuis et vers d'autres types de données, à quelques exceptions près. Pour de plus amples informations, veuillez consulter Limites.

Séquences de classement

Amazon Redshift ne prend pas en charge les séquences de classement spécifiques à une région ou définies par l'utilisateur. En général, les résultats d'un prédicat dans quelque contexte que ce soit peuvent être affectés par l'absence de règles spécifiques aux paramètres régionaux en matière de tri et de comparaison des valeurs de données. Par exemple, les expressions et fonctions ORDER BY telles que MIN, MAX et RANK renvoient des résultats basés sur un UTF8 ordre binaire des données qui ne prend pas en compte les caractères spécifiques aux paramètres régionaux.

Expressions

Rubriques

- Expressions simples
- Expressions composées
- Listes d'expressions

- Sous-requêtes scalaires
- Expressions de fonction

Une expression est une combinaison d'un(e) ou de plusieurs valeurs, opérateurs ou fonctions qui correspondent à une valeur. Le type de données d'une expression est généralement celui de ses composants.

Expressions simples

Une expression simple est l'une des expressions suivantes :

- Une constante ou une valeur littérale
- Un nom de colonne ou une référence de colonne
- Une fonction scalaire
- Une fonction d'agrégation (ensemble)
- Une fonction de fenêtre
- Une sous-requête scalaire

Exemples d'expressions simples :

```
5+12
dateid
sales.qtysold * 100
sqrt (4)
max (qtysold)
(select max (qtysold) from sales)
```

Expressions composées

Une expression composée est une série d'expressions simples jointes par des opérateurs arithmétiques. Une expression simple utilisée dans une expression composée doit retourner une valeur numérique.

Syntaxe

```
expression
operator
```

```
expression | (compound_expression)
```

Arguments

expression

Une expression simple qui correspond à une valeur.

opérateur

Une expression arithmétique composée peut être construite à l'aide des opérateurs suivants, en respectant l'ordre de priorité :

- () : parenthèses pour contrôler l'ordre de l'évaluation
- +,- : signe/opérateur positif et négatif
- ^ , |/ , || / : élévation à la puissance, racine carrée, racine cubique
- *, /, % : multiplication, division et opérateurs modulo
- @ : valeur absolue
- +,- : addition et soustraction
- & , |, #, ~, <<, >> : AND, OR, NOT, opérateurs au niveau du bit de décalage gauche, décalage droit
- || : concaténation

(expression_composée)

Les expressions composées peuvent être imbriquées à l'aide de parenthèses.

Exemples

Voici quelques exemples d'expressions composées.

```
('SMITH' || 'JONES')
sum(x) / y
sqrt(256) * avg(column)
rank() over (order by qtysold) / 100
(select (pricepaid - commission) from sales where dateid = 1882) * (qtysold)
```

Certaines fonctions peuvent également être imbriquées dans d'autres fonctions. Par exemple, une fonction scalaire peut être imbriquée dans une autre fonction scalaire. L'exemple suivant retourne la somme des valeurs absolues d'un ensemble de nombres :

```
sum(abs(qtysold))
```

Les fonctions de fenêtrage ne peuvent pas être utilisées comme arguments pour les fonctions d'agrégation ou d'autres fonctions de fenêtrage. L'expression suivante retourne une erreur :

```
avg(rank() over (order by qtysold))
```

Les fonctions de fenêtrage peuvent avoir une fonction d'agrégation imbriquée. L'expression suivante additionne des ensembles de valeurs, puis les classe :

```
rank() over (order by sum(qtysold))
```

Listes d'expressions

Une liste d'expressions est une combinaison d'expressions et peut apparaître dans les conditions d'appartenance et de comparaison (clauses WHERE) et dans les clauses GROUP BY.

Syntaxe

```
expression , expression , ... | (expression, expression, ...)
```

Arguments

expression

Une expression simple qui correspond à une valeur. Une liste d'expressions peut contenir une ou plusieurs expressions séparées par des virgules, ou un ou plusieurs ensembles d'expressions séparés par des virgules. Lorsqu'il existe plusieurs ensembles d'expressions, chaque ensemble doit comporter le même nombre d'expressions et être séparée par des parenthèses. Le nombre d'expressions de chaque ensemble doit correspondre au nombre d'expressions avant l'opérateur de la condition.

Exemples

Exemples de listes d'expressions dans des conditions :

```
(1, 5, 10)
('THESE', 'ARE', 'STRINGS')
(('one', 'two', 'three'), ('blue', 'yellow', 'green'))
```

Le nombre d'expressions de chaque ensemble doit correspondre au nombre dans la première partie de l'instruction :

```
select * from venue
where (venuecity, venuestate) in (('Miami', 'FL'), ('Tampa', 'FL'))
order by venueid;

venueid |        venuename        | venuecity | venuestate | venueseats
---------+-------------------------+-----------+------------+------------
28 | American Airlines Arena | Miami     | FL         |          0
54 | St. Pete Times Forum    | Tampa     | FL         |          0
91 | Raymond James Stadium   | Tampa     | FL         |      65647
(3 rows)
```

Sous-requêtes scalaires

Une sous-requête scalaire est une requête SELECT régulière entre parenthèses qui renvoie exactement une valeur : une seule ligne avec une colonne. La requête est exécutée et la valeur retournée est utilisée dans la requête externe. Si la sous-requête ne renvoie aucune ligne, l'expression de la sous-requête a la valeur null. Si elle renvoie plusieurs lignes, Amazon Redshift renvoie une erreur. La sous-requête peut faire référence aux variables de la requête parente, qui serviront de constantes lors d'une invocation de la sous-requête.

Vous pouvez utiliser des sous-requêtes scalaires dans la plupart des instructions qui appellent une expression. Les sous-requêtes scalaires ne sont pas des expressions valides dans les cas suivants :

- Comme valeurs par défaut pour les expressions
- Dans les clauses GROUP BY et HAVING

exemple

La sous-requête suivante calcule le prix moyen payé par vente sur toute l'année 2008, puis la requête externe utilise cette valeur dans la sortie pour la comparer au prix moyen par vente par trimestre :

```
select qtr, avg(pricepaid) as avg_saleprice_per_qtr,
(select avg(pricepaid)
from sales join date on sales.dateid=date.dateid
where year = 2008) as avg_saleprice_yearly
from sales join date on sales.dateid=date.dateid
where year = 2008
```

```
group by qtr
order by qtr;
qtr  | avg_saleprice_per_qtr | avg_saleprice_yearly
-------+----------------------+---------------------
1     |                647.64 |               642.28
2     |                646.86 |               642.28
3     |                636.79 |               642.28
4     |                638.26 |               642.28
(4 rows)
```

Expressions de fonction

Syntaxe

Toute fonction intégrée peut être utilisée en tant qu'expression. La syntaxe d'un appel de fonction est le nom d'une fonction suivi de sa liste d'arguments entre parenthèses.

```
function ( [expression [, expression...]] )
```

Arguments

fonction

Toute fonction intégrée. Pour des exemples de fonctions, consultez Référence sur les fonctions SQL.

expression

Toute expression correspondant au type de données et au nombre de paramètres attendu par la fonction.

Exemples

```
abs (variable)
select avg (qtysold + 3) from sales;
select dateadd (day,30,caldate) as plus30days from date;
```

Conditions

Rubriques

- Syntaxe

- Condition de comparaison
- Conditions logiques
- Conditions de correspondance de modèles
- Condition de plage BETWEEN
- Condition null
- Condition EXISTS
- Condition IN

Une condition est une instruction d'un(e) ou plusieurs expressions et opérateurs logiques qui ont la valeur true, false ou unknown. Les conditions sont également appelées parfois prédicats.

> **Note**
>
> Toutes les comparaisons de chaîne et correspondances du modèle LIKE sont sensibles à la casse. Par exemple, « A » et « a » ne correspondent pas. Cependant, vous pouvez effectuer une correspondance de modèle non sensible à la casse à l'aide du prédicat ILIKE.

Syntaxe

```
comparison_condition
| logical_condition
| range_condition
| pattern_matching_condition
| null_condition
| EXISTS_condition
| IN_condition
```

Condition de comparaison

Les conditions de comparaison établissent des relations logiques entre deux valeurs. Toutes les conditions de comparaison sont des opérateurs binaires avec un type de retour booléen. Amazon Redshift prend en charge les opérateurs de comparaison décrits dans le tableau suivant :

Opérateur	Syntaxe	Description
<	a < b	La valeur a est inférieure à la valeur b.

Opérateur	Syntaxe	Description
>	`a > b`	La valeur a est supérieure à la valeur b.
<=	`a <= b`	La valeur a est inférieure ou égale à la valeur b.
>=	`a >= b`	La valeur a est supérieure ou égale à la valeur b.
=	`a = b`	La valeur a est égale à la valeur b.
<> ou !=	`a <> b or a != b`	La valeur a n'est pas égale à la valeur b.
ANY | SOME	`a = ANY(subquery)`	La valeur a est égale à une valeur retournée par la sous-requête.
ALL	`a <> ALL or != ALL (subquery))`	La valeur a n'est pas égale à une valeur retournée par la sous-requête.
IS TRUE | FALSE | UNKNOWN	`a IS TRUE`	La valeur a la valeur booléenne TRUE.

Notes d'utilisation

= ANY | SOME

Les mots-clés ANY et SOME sont synonymes de la condition IN, et renvoient true (vrai) si la comparaison est vraie pour au moins une valeur renvoyée par une sous-requête qui renvoie une ou plusieurs valeurs. Amazon Redshift prend en charge uniquement la condition = (égal) pour ANY et SOME. Les conditions d'inégalité ne sont pas prises en charge.

> **Note**
>
> Le prédicat ALL n'est pas pris en charge.

<> ALL

Le mot-clé ALL est synonyme de NOT IN (voir condition Condition IN) et renvoie true (vrai) si l'expression n'est pas incluse dans les résultats de la sous-requête. Amazon Redshift prend

en charge uniquement la condition <> ou != (non égal) pour ALL. Les autres conditions de comparaison ne sont pas prises en charge.

EST TRUE/FALSE/UNKNOWN

Les valeurs différentes de zéro correspondent à TRUE, 0 correspond à FALSE et null équivaut à UNKNOWN. Consultez le type de données Type Boolean.

Exemples

Voici quelques exemples simples de conditions de comparaison :

```
a = 5
a < b
min(x) >= 5
qtysold = any (select qtysold from sales where dateid = 1882
```

La requête suivante retourne les sites de plus de 10 000 places à partir de la table VENUE :

```
select venueid, venuename, venueseats from venue
where venueseats > 10000
order by venueseats desc;

venueid |           venuename            | venueseats
---------+--------------------------------+------------
83 | FedExField                     |      91704
 6 | New York Giants Stadium        |      80242
79 | Arrowhead Stadium              |      79451
78 | INVESCO Field                  |      76125
69 | Dolphin Stadium                |      74916
67 | Ralph Wilson Stadium           |      73967
76 | Jacksonville Municipal Stadium |      73800
89 | Bank of America Stadium        |      73298
72 | Cleveland Browns Stadium       |      73200
86 | Lambeau Field                  |      72922
...
(57 rows)
```

Cet exemple sélectionne les utilisateurs (USERID) de la table USERS qui aiment la musique rock :

```
select userid from users where likerock = 't' order by 1 limit 5;
```

```
userid
--------
3
5
6
13
16
(5 rows)
```

Cet exemple sélectionne les utilisateurs (USERID) de la table USERS pour lesquels on ignore s'ils aiment la musique rock :

```
select firstname, lastname, likerock
from users
where likerock is unknown
order by userid limit 10;

firstname | lastname | likerock
----------+----------+----------
Rafael    | Taylor   |
Vladimir  | Humphrey |
Barry     | Roy      |
Tamekah   | Juarez   |
Mufutau   | Watkins  |
Naida     | Calderon |
Anika     | Huff     |
Bruce     | Beck     |
Mallory   | Farrell  |
Scarlett  | Mayer    |
(10 rows
```

Exemples avec une colonne TIME

La table d'exemple TIME_TEST suivante comporte une colonne TIME_VAL (type TIME) dans laquelle trois valeurs ont été insérées.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
```

```
00:58:00
```

L'exemple suivant extrait les heures de chaque timetz_val.

```
select time_val from time_test where time_val < '3:00';
   time_val
---------------
 00:00:00.5550
 00:58:00
```

L'exemple suivant compare deux littéraux de type heure.

```
select time '18:25:33.123456' = time '18:25:33.123456';
 ?column?
----------
 t
```

Exemples avec une colonne TIMETZ

L'exemple de tableau TIMETZ_TEST suivant comporte une colonne TIMETZ_VAL (type TIMETZ) dans laquelle trois valeurs ont été insérées.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

L'exemple suivant sélectionne uniquement les valeurs TIMETZ inférieures à `3:00:00 UTC`. La comparaison est effectuée après la conversion de la valeur en UTC.

```
select timetz_val from timetz_test where timetz_val < '3:00:00 UTC';

   timetz_val
---------------
 00:00:00.5550+00
```

L'exemple suivant compare deux littéraux TIMETZ. Le fuseau horaire n'est pas pris en compte pour la comparaison.

```
select time '18:25:33.123456 PST' < time '19:25:33.123456 EST';

 ?column?
----------
 t
```

Conditions logiques

Les conditions logiques combinent le résultat de deux conditions pour produire un résultat unique. Toutes les conditions logiques sont des opérateurs binaires avec un type de retour booléen.

Syntaxe

```
expression
{ AND | OR }
expression
NOT expression
```

Les conditions logiques utilisent une logique booléenne à trois valeurs où la valeur nulle représente une relation inconnue. Le tableau suivant décrit les résultats des conditions logiques, où E1 et E2 représentent des expressions :

E1	E2	E1 AND E2	E1 OR E2	NOT E2
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	TRUE
TRUE	UNKNOWN	UNKNOWN	TRUE	UNKNOWN
FALSE	TRUE	FALSE	TRUE	
FALSE	FALSE	FALSE	FALSE	
FALSE	UNKNOWN	FALSE	UNKNOWN	
UNKNOWN	TRUE	UNKNOWN	TRUE	
UNKNOWN	FALSE	FALSE	UNKNOWN	
UNKNOWN	UNKNOWN	UNKNOWN	UNKNOWN	

L'opérateur NOT est analysé avant AND et l'opérateur AND est évalué avant l'opérateur OR. Les parenthèses utilisées peuvent remplacer cet ordre d'évaluation par défaut.

Exemples

L'exemple suivant retourne USERID et USERNAME de la table USERS où l'utilisateur aime à la fois Las Vegas et les sports :

```
select userid, username from users
where likevegas = 1 and likesports = 1
order by userid;

userid | username
--------+----------
1 | JSG99FHE
67 | TWU10MZT
87 | DUF19VXU
92 | HYP36WEQ
109 | FPL38HZK
120 | DMJ24GUZ
123 | QZR22XGQ
130 | ZQC82ALK
133 | LBN45WCH
144 | UCX04JKN
165 | TEY68OEB
169 | AYQ83HGO
184 | TVX65AZX
...
(2128 rows)
```

L'exemple suivant retourne USERID et USERNAME de la table USERS où l'utilisateur aime Las Vegas, ou les sports, ou les deux. Cette requête renvoie toutes les données de sortie de l'exemple précédent, plus les utilisateurs qui aiment uniquement Las Vegas ou le sport.

```
select userid, username from users
where likevegas = 1 or likesports = 1
order by userid;

userid | username
--------+----------
1 | JSG99FHE
2 | PGL08LJI
```

```
3 | IFT66TXU
5 | AEB55QTM
6 | NDQ15VBM
9 | MSD36KVR
10 | WKW41AIW
13 | QTF33MCG
15 | OWU78MTR
16 | ZMG93CDD
22 | RHT62AGI
27 | KOY02CVE
29 | HUH27PKK
...
(18968 rows)
```

La requête suivante utilise des parenthèses autour de la condition OR pour trouver les salles de New York ou de Californie où Macbeth a été joué :

```
select distinct venuename, venuecity
from venue join event on venue.venueid=event.venueid
where (venuestate = 'NY' or venuestate = 'CA') and eventname='Macbeth'
order by 2,1;

venuename                |   venuecity
-----------------------------------------+---------------
Geffen Playhouse                        | Los Angeles
Greek Theatre                           | Los Angeles
Royce Hall                              | Los Angeles
American Airlines Theatre               | New York City
August Wilson Theatre                   | New York City
Belasco Theatre                         | New York City
Bernard B. Jacobs Theatre               | New York City
...
```

La suppression des parenthèses de cet exemple modifie la logique et les résultats de la requête.

Les exemples suivants utilisent l'opérateur NOT :

```
select * from category
where not catid=1
order by 1;

catid | catgroup |  catname  |                 catdesc
```

```
-------+----------+-----------+--------------------------------------------
2 | Sports   | NHL       | National Hockey League
3 | Sports   | NFL       | National Football League
4 | Sports   | NBA       | National Basketball Association
5 | Sports   | MLS       | Major League Soccer
...
```

L'exemple suivant utilise une condition NOT suivie d'une condition AND :

```
select * from category
where (not catid=1) and catgroup='Sports'
order by catid;

catid | catgroup | catname |             catdesc
-------+----------+---------+---------------------------------
2 | Sports   | NHL     | National Hockey League
3 | Sports   | NFL     | National Football League
4 | Sports   | NBA     | National Basketball Association
5 | Sports   | MLS     | Major League Soccer
(4 rows)
```

Conditions de correspondance de modèles

Rubriques

- LIKE
- SIMILAR TO
- Opérateurs POSIX

Un opérateur de correspondance de modèle recherche une chaîne d'un modèle spécifié dans l'expression conditionnelle et retourne true ou false selon qu'il a trouvé ou non une correspondance. Amazon Redshift utilise trois méthodes pour la correspondance des modèles :

- Expressions LIKE

  L'opérateur LIKE compare une expression de chaîne, comme un nom de colonne, avec un modèle qui utilise les caractères génériques % (pourcentage) et _ (soulignement). La correspondance de modèle LIKE couvre toute la chaîne. LIKE effectue une correspondance sensible à la casse et ILIKE effectue une correspondance non sensible à la casse.
- Expressions régulières SIMILAR TO

L'opérateur SIMILAR TO correspond à une expression de chaîne avec un modèle d'expression régulière SQL standard, ce qui peut inclure un ensemble de métacaractères de correspondance de modèle incluant les deux pris en charge par l'opérateur LIKE. SIMILAR TO correspond à la totalité de la chaîne et effectue une correspondance sensible à la casse.

- Expressions régulières POSIX

  Les expressions régulières POSIX fournissent un moyen plus puissant pour la correspondance de modèles que les opérateurs LIKE et SIMILAR TO. Les modèles d'expressions régulières POSIX peuvent correspondre à n'importe quelle partie de la chaîne et effectuent une correspondance sensible à la casse.

La correspondance d'expressions régulières, à l'aide des opérateurs SIMILAR TO ou POSIX, est coûteuse en termes de calcul. Nous vous conseillons d'utiliser LIKE autant que possible, notamment lors du traitement d'un très grand nombre de lignes. Par exemple, les requêtes suivantes sont fonctionnellement identiques, mais la requête qui utilise LIKE s'exécute infiniment plus vite que la requête qui utilise une expression régulière :

```
select count(*) from event where eventname SIMILAR TO '%(Ring|Die)%';
select count(*) from event where eventname LIKE '%Ring%' OR eventname LIKE '%Die%';
```

LIKE

L'opérateur LIKE compare une expression de chaîne, comme un nom de colonne, avec un modèle qui utilise les caractères génériques % (pourcentage) et _ (soulignement). La correspondance de modèle LIKE couvre toute la chaîne. Pour faire correspondre une séquence à n'importe quel emplacement au sein d'une chaîne, le modèle doit commencer et finir par un signe %.

LIKE est sensible à la casse, ILIKE ne l'est pas.

Syntaxe

```
expression [ NOT ] LIKE | ILIKE pattern [ ESCAPE 'escape_char' ]
```

Arguments

expression

Expression de caractère UTF-8 valide, comme un nom de colonne.

LIKE | ILIKE

LIKE effectue une correspondance sensible à la casse. ILIKE effectue une correspondance de modèle non sensible à la casse pour les caractères UTF-8 (ASCII) codés sur un octet. Pour effectuer une correspondance de modèle non sensible à la casse pour les caractères codés sur plusieurs octets, utilisez la fonction LOWER sur expression et pattern avec une condition LIKE.

Contrairement aux prédicats de comparaison, tels que = et <>, les prédicats LIKE et ILIKE n'ignorent pas implicitement les espaces de fin. Pour ignorer les espaces de fin, utilisez RTRIM ou convertissez explicitement une colonne CHAR en VARCHAR.

L'opérateur ~~ est équivalent à LIKE et ~~* est équivalent à ILIKE. Les opérateurs !~~ et !~~* sont également équivalents à NOT LIKE et NOT ILIKE.

pattern

Expression de caractère UTF-8 valide avec le modèle à mettre en correspondance.

escape_char

Expression de caractère qui utilise une séquence d'échappement pour les méta-caractères du modèle. La valeur par défaut est deux barres obliques inverses ('\\ »).

Si pattern ne contient pas de méta-caractères, le modèle représente uniquement la chaîne elle-même ; dans ce cas, LIKE agit de même que l'opérateur d'égalité.

Les expressions de caractère peuvent avoir CHAR ou VARCHAR comme type de données. En cas de différence, Amazon Redshift convertit pattern au type de données de l'expression.

LIKE prend en charge les méta-caractères de correspondance de modèle suivants :

Opérateur	Description
%	Met en correspondance une séquence de zéro ou plusieurs caractères.
_	Met en correspondance un seul caractère.

Exemples

Le tableau suivant montre des exemples de correspondance de modèle avec LIKE :

Expression	Renvoie
`'abc' LIKE 'abc'`	True
`'abc' LIKE 'a%'`	True
`'abc' LIKE '_B_'`	False
`'abc' ILIKE '_B_'`	True
`'abc' LIKE 'c%'`	False

L'exemple suivant recherche toutes les villes dont le nom commence par « E » :

```
select distinct city from users
where city like 'E%' order by city;
city
---------------
East Hartford
East Lansing
East Rutherford
East St. Louis
Easthampton
Easton
Eatontown
Eau Claire
...
```

L'exemple suivant recherche les utilisateurs dont le nom contient « ten » :

```
select distinct lastname from users
where lastname like '%ten%' order by lastname;
lastname
-------------
Christensen
Wooten
...
```

L'exemple suivant montre comment associer plusieurs modèles.

```
select distinct lastname from tickit.users
where lastname like 'Chris%' or lastname like '%Wooten' order by lastname;
lastname
-------------
Christensen
Christian
Wooten
...
```

L'exemple suivant recherche villes dont les troisième et quatrième caractères sont « ea ». La commande utilise ILIKE pour démontrer l'insensibilité à la casse :

```
select distinct city from users where city ilike '__EA%' order by city;
city
-------------
Brea
Clearwater
Great Falls
Ocean City
Olean
Wheaton
(6 rows)
```

L'exemple suivant utilise la chaîne d'échappement par défaut (\\) pour rechercher les chaînes qui incluent « start_» (texte `start` suivi d'un trait de soulignement _) :

```
select tablename, "column" from pg_table_def
where "column" like '%start\\_%'
limit 5;

     tablename     |    column
-------------------+---------------
 stl_s3client      | start_time
 stl_tr_conflict   | xact_start_ts
 stl_undone        | undo_start_ts
 stl_unload_log    | start_time
 stl_vacuum_detail | start_row
(5 rows)
```

L'exemple suivant spécifie « ^ » comme caractère d'échappement, puis utilise ce dernier pour rechercher des chaînes qui incluent « start_ » (texte `start` suivi d'un trait de soulignement _) :

```
select tablename, "column" from pg_table_def
where "column" like '%start^_%' escape '^'
limit 5;

     tablename     |    column
-------------------+---------------
 stl_s3client      | start_time
 stl_tr_conflict   | xact_start_ts
 stl_undone        | undo_start_ts
 stl_unload_log    | start_time
 stl_vacuum_detail | start_row
(5 rows)
```

L'exemple suivant utilise l'opérateur ~~* pour effectuer une recherche non sensible à la casse (ILIKE) pour les villes commençant par « Ag ».

```
select distinct city from users where city ~~* 'Ag%' order by city;

city
------------
Agat
Agawam
Agoura Hills
Aguadilla
```

SIMILAR TO

L'opérateur SIMILAR TO met en correspondance une expression de chaîne, comme un nom de colonne, avec un modèle d'expression régulière SQL standard. Un modèle d'expression régulière SQL peut inclure un ensemble de méta-caractères de correspondance de modèle, y compris les deux pris en charge par l'opérateur LIKE.

L'opérateur SIMILAR TO retourne true uniquement si son modèle correspond à l'ensemble de la chaîne, à la différence du comportement de l'expression régulière POSIX, où le modèle peut correspondre à n'importe quelle partie de la chaîne.

SIMILAR TO effectue une correspondance sensible à la casse.

> **Note**
>
> La correspondance d'expression régulière à l'aide de SIMILAR TO est coûteuse en termes de calcul. Nous vous conseillons d'utiliser LIKE autant que possible, notamment lors du traitement d'un très grand nombre de lignes. Par exemple, les requêtes suivantes sont fonctionnellement identiques, mais la requête qui utilise LIKE s'exécute infiniment plus vite que la requête qui utilise une expression régulière :
>
> ```
> select count(*) from event where eventname SIMILAR TO '%(Ring|Die)%';
> select count(*) from event where eventname LIKE '%Ring%' OR eventname LIKE '%Die
> %';
> ```

Syntaxe

```
expression [ NOT ] SIMILAR TO pattern [ ESCAPE 'escape_char' ]
```

Arguments

expression

Expression de caractère UTF-8 valide, comme un nom de colonne.

SIMILAR TO

SIMILAR TO effectue une correspondance sensible à la casse pour toute la chaîne de l'expression.

pattern

Expression de caractères UTF-8 valide représentant un modèle d'expression régulière SQL standard.

escape_char

Expression de caractères qui utilise une séquence d'échappement pour les méta-caractères du modèle. La valeur par défaut est deux barres obliques inverses ('\\ »).

Si pattern ne contient pas de méta-caractères, le modèle représente uniquement la chaîne elle-même.

Les expressions de caractère peuvent avoir CHAR ou VARCHAR comme type de données. En cas de différence, Amazon Redshift convertit pattern au type de données de l'expression.

SIMILAR TO prend en charge les méta-caractères de correspondance de modèle suivants :

Opérateur	Description
`%`	Met en correspondance une séquence de zéro ou plusieurs caractères.
`_`	Met en correspondance un seul caractère.
`|`	Indique une alternative (l'une ou l'autre des deux possibilités).
`*`	Répétez l'élément précédent zéro ou plusieurs fois.
`+`	Répétez l'élément précédent une ou plusieurs fois.
`?`	Répétez l'élément précédent zéro ou une fois.
`{m}`	Répétez l'élément précédent exactement m fois.
`{m,}`	Répétez l'élément précédent m ou plusieurs fois.
`{m,n}`	Répétez l'élément précédent au moins m fois et pas plus de n fois.
`()`	Placez entre parenthèses les éléments d'un groupe sous forme d'un seul élément logique.
`[...]`	Une expression entre crochets spécifie une classe de caractères, comme dans les expressions régulières POSIX.

Exemples

Le tableau suivant illustre des exemples de correspondance de modèle à l'aide de SIMILAR TO :

Expression	Renvoie
`'abc' SIMILAR TO 'abc'`	True
`'abc' SIMILAR TO '_b_'`	True

Expression	Renvoie
`'abc' SIMILAR TO '_A_'`	False
`'abc' SIMILAR TO '%(b|d)%'`	True
`'abc' SIMILAR TO '(b|c)%'`	False
`'AbcAbcdefgefg12efgefg12' SIMILAR TO '((Ab)?c)+d((efg)+(12))+'`	True
`'aaaaaab11111xy' SIMILAR TO 'a{6}_[0-9]{5}(x|y){2}'`	True
`'$0.87' SIMILAR TO '$[0-9]+(.[0-9][0-9])?'`	True

L'exemple suivant recherche toutes les villes dont le nom contient « E » ou « H » :

```
SELECT DISTINCT city FROM users
WHERE city SIMILAR TO '%E%|%H%' ORDER BY city LIMIT 5;

      city
-----------------
 Agoura Hills
 Auburn Hills
 Benton Harbor
 Beverly Hills
 Chicago Heights
```

L'exemple suivant utilise la chaîne d'échappement par défaut (« \\ ») pour rechercher les chaînes qui incluent « _ » :

```
SELECT tablename, "column" FROM pg_table_def
WHERE "column" SIMILAR TO '%start\\_%'
ORDER BY tablename, "column" LIMIT 5;

        tablename         |       column
--------------------------+---------------------
 stcs_abort_idle          | idle_start_time
```

```
 stcs_abort_idle          | txn_start_time
 stcs_analyze_compression | start_time
 stcs_auto_worker_levels  | start_level
 stcs_auto_worker_levels  | start_wlm_occupancy
```

Les exemples suivants spécifient « ^ » comme caractère d'échappement, puis utilise le caractère d'échappement pour rechercher des chaînes qui incluent « _ » :

```
SELECT tablename, "column" FROM pg_table_def
WHERE "column" SIMILAR TO '%start^_%' ESCAPE '^'
ORDER BY tablename, "column" LIMIT 5;

        tablename         |       column
--------------------------+--------------------
 stcs_abort_idle          | idle_start_time
 stcs_abort_idle          | txn_start_time
 stcs_analyze_compression | start_time
 stcs_auto_worker_levels  | start_level
 stcs_auto_worker_levels  | start_wlm_occupancy
```

Opérateurs POSIX

Une expression régulière POSIX est une séquence de caractères qui spécifie un modèle de correspondance. Une chaîne correspond à une expression régulière si elle fait partie du jeu régulier décrit par l'expression régulière.

Les expressions régulières POSIX fournissent un moyen plus puissant pour la correspondance de modèle que les opérateurs LIKE et SIMILAR TO. Les modèles d'expressions régulières POSIX peuvent correspondre à une partie quelconque d'une chaîne, contrairement à l'opérateur SIMILAR TO, qui retourne true uniquement si son modèle correspond à la totalité de la chaîne.

> **Note**
>
> La correspondance d'expressions régulières à l'aide des opérateurs POSIX est coûteuse en termes de calcul. Nous vous conseillons d'utiliser LIKE autant que possible, notamment lors du traitement d'un très grand nombre de lignes. Par exemple, les requêtes suivantes sont fonctionnellement identiques, mais la requête qui utilise LIKE s'exécute infiniment plus vite que la requête qui utilise une expression régulière :
>
> ```
> select count(*) from event where eventname ~ '.*(Ring|Die).*';
> ```

```
select count(*) from event where eventname LIKE '%Ring%' OR eventname LIKE '%Die
%';
```

Syntaxe

```
expression [ ! ] ~ pattern
```

Arguments

expression

Expression de caractère UTF-8 valide, comme un nom de colonne.

!

Opérateur de négation. Ne correspond pas à l'expression régulière.

~

Effectuez une correspondance sensible à la casse pour une sous-chaîne d'expression.

> **Note**
>
> ~~ est un synonyme pour LIKE.

pattern

Chaîne littérale qui représente un modèle d'expression régulière.

Si pattern ne contient pas de caractères génériques, le modèle représente uniquement la chaîne elle-même.

Pour rechercher des chaînes qui incluent des méta-caractères, tels que « `. * | ?` », et ainsi de suite, faites précéder le caractère de la séquence d'échappement composée de deux barres obliques inverses « `\\` »). Contrairement à `SIMILAR TO` et `LIKE`, la syntaxe des expressions régulières POSIX ne gère pas de caractère d'échappement défini par l'utilisateur.

Les expressions de caractère peuvent avoir CHAR ou VARCHAR comme type de données. En cas de différence, Amazon Redshift convertit pattern au type de données de l'expression.

Toutes les expressions de caractères peuvent avoir CHAR ou VARCHAR comme type de données. Si les expressions diffèrent par le type de données, Amazon Redshift les convertit au type de données de l'expression.

Le modèle de correspondance POSIX prend en charge les méta-caractères suivants :

POSIX	Description
.	Met en correspondance un seul caractère.
*	Correspond à zéro ou plusieurs occurrences.
+	Correspond à une ou plusieurs occurrences.
?	Correspond à zéro ou une occurrence.
|	Spécifie d'autres correspondances ; par exemple, E | H signifie E ou H.
^	Correspond au beginning-of-line personnage.
$	Correspond au end-of-line personnage.
$	Correspond à la fin de la chaîne.
[]	Les crochets spécifient une liste de correspondance, qui doit correspon dre à une expression de la liste. L'accent circonflexe (^) précède une liste de non-correspondance, qui correspond à tout caractère à l'exception des expressions représentées dans la liste.
( )	Placez entre parenthèses les éléments d'un groupe sous forme d'un seul élément logique.
{m}	Répétez l'élément précédent exactement m fois.
{m,}	Répétez l'élément précédent m ou plusieurs fois.
{m,n}	Répétez l'élément précédent au moins m fois et pas plus de n fois.
[: :]	Correspond à tout caractère au sein d'une classe de caractères POSIX. Dans les classes de caractères suivantes, Amazon Redshift ne prend

POSIX	Description
	en charge que les caractères ASCII : `[:alnum:]` , `[:alpha:]` , `[:lower:]` , `[:upper:]`

Amazon Redshift prend en charge les classes de caractères POSIX suivantes.

Classe de caractères	Description
`[[:alnum:]]`	Tous les caractères ASCII alphanumériques
`[[:alpha:]]`	Tous les caractères ASCII alphabétiques
`[[:blank:]]`	Tous les caractères espace
`[[:cntrl:]]`	Tous les caractères de contrôle (non affichables)
`[[:digit:]]`	Tous les chiffres numériques
`[[:lower:]]`	Tous les caractères ASCII alphabétiques minuscules
`[[:punct:]]`	Tous les caractères de ponctuation
`[[:space:]]`	Tous les caractères espace (non affichables)
`[[:upper:]]`	Tous les caractères ASCII alphabétiques majuscules
`[[:xdigit:]]`	Tous les caractères hexadécimaux valides

Amazon Redshift prend en charge les opérateurs suivants, influencés par Perl, dans les expressions régulières. Échappez l'opérateur à l'aide de deux barres obliques inverses (« »). ('\\').

Opérateur	Description	Expression de classe de caractères équivalente
\\d	Un caractère numérique	`[[:digit:]]`

Opérateur	Description	Expression de classe de caractères équivalente
\\D	Un caractère non numérique	[^[:digit:]]
\\w	Un caractère de mot	[[:word:]]
\\W	Un caractère autre qu'un caractère de mot	[^[:word:]]
\\s	Un caractère espace blanc	[[:space:]]
\\S	Un caractère autre qu'un caractère espace blanc	[^[:space:]]
\\b	Un mot limite	

Exemples

Le tableau suivant montre des exemples de correspondance de modèle à l'aide des opérateurs POSIX :

Expression	Renvoie
'abc' ~ 'abc'	True
'abc' ~ 'a'	True
'abc' ~ 'A'	False
'abc' ~ '.*(b|d).*'	True
'abc' ~ '(b|c).*'	True
'AbcAbcdefgefg12efgefg12' ~ '((Ab)?c)+d((efg)+(12))+'	True
'aaaaaab11111xy' ~ 'a{6}.[1]{5}(x|y){2}'	True

Expression	Renvoie
`'$0.87' ~ '\\$[0-9]+(\\.[0-9][0-9])?'`	True
`'ab c' ~ '[[:space:]]'`	True
`'ab c' ~ '\\s'`	True
`' ' ~ '\\S'`	False

L'exemple suivant recherche toutes les villes dont le nom contient E ou H :

```
SELECT DISTINCT city FROM users
WHERE city ~ '.*E.*|.*H.*' ORDER BY city LIMIT 5;

      city
-----------------
 Agoura Hills
 Auburn Hills
 Benton Harbor
 Beverly Hills
 Chicago Heights
```

L'exemple suivant recherche toutes les villes dont le nom ne contient pas E ou H :

```
SELECT DISTINCT city FROM users WHERE city !~ '.*E.*|.*H.*' ORDER BY city LIMIT 5;

      city
-----------------
 Aberdeen
 Abilene
 Ada
 Agat
 Agawam
```

L'exemple suivant utilise la chaîne d'échappement (« \\ ») pour rechercher les chaînes qui incluent un point.

```
SELECT venuename FROM venue
```

```
WHERE venuename ~ '.*\\..*'
ORDER BY venueid;

          venuename
------------------------------
 St. Pete Times Forum
 Jobing.com Arena
 Hubert H. Humphrey Metrodome
 U.S. Cellular Field
 Superpages.com Center
 E.J. Nutter Center
 Bernard B. Jacobs Theatre
 St. James Theatre
```

Condition de plage BETWEEN

Une condition BETWEEN teste les expressions pour l'inclusion dans une plage de valeurs, à l'aide des mots-clés BETWEEN et AND.

Syntaxe

```
expression [ NOT ] BETWEEN expression AND expression
```

Les expressions peuvent être de type de données numeric, character ou datetime, mais elles doivent être compatibles. La plage est inclusive.

Exemples

Le premier exemple comptabilise le nombre de transactions ayant enregistré des ventes de 2, 3 ou 4 billets :

```
select count(*) from sales
where qtysold between 2 and 4;

count
--------
104021
(1 row)
```

La condition de la plage comprend les valeurs de début et de fin.

```
select min(dateid), max(dateid) from sales
where dateid between 1900 and 1910;

min  | max
-----+-----
1900 | 1910
```

La première expression d'une condition de plage doit être la valeur inférieure et la deuxième expression la valeur supérieure. L'exemple suivant retourne toujours zéro ligne en raison des valeurs des expressions :

```
select count(*) from sales
where qtysold between 4 and 2;

count
-------
0
(1 row)
```

Cependant, l'application du modificateur NOT inverse la logique et génère le nombre de toutes les lignes :

```
select count(*) from sales
where qtysold not between 4 and 2;

count
--------
172456
(1 row)
```

La requête suivante retourne une liste des salles avec 20 000 à 50 000 places :

```
select venueid, venuename, venueseats from venue
where venueseats between 20000 and 50000
order by venueseats desc;

venueid |       venuename               | venueseats
---------+-------------------------------+------------
116 | Busch Stadium                 |      49660
106 | Rangers BallPark in Arlington |      49115
96 | Oriole Park at Camden Yards   |      48876
```

```
...
(22 rows)
```

L'exemple suivant illustre l'utilisation de BETWEEN pour les valeurs de date :

```
select salesid, qtysold, pricepaid, commission, saletime
from sales
where eventid between 1000 and 2000
   and saletime between '2008-01-01' and '2008-01-03'
order by saletime asc;

salesid | qtysold | pricepaid | commission |   saletime
--------+---------+-----------+------------+---------------
  65082 |       4 |       472 |       70.8 | 1/1/2008 06:06
 110917 |       1 |       337 |      50.55 | 1/1/2008 07:05
 112103 |       1 |       241 |      36.15 | 1/2/2008 03:15
 137882 |       3 |      1473 |     220.95 | 1/2/2008 05:18
  40331 |       2 |        58 |        8.7 | 1/2/2008 05:57
 110918 |       3 |      1011 |     151.65 | 1/2/2008 07:17
  96274 |       1 |       104 |       15.6 | 1/2/2008 07:18
 150499 |       3 |       135 |      20.25 | 1/2/2008 07:20
  68413 |       2 |       158 |       23.7 | 1/2/2008 08:12
```

Notez que même si la plage de BETWEEN est inclusive, les dates ont par défaut une valeur horaire de 00:00:00. La seule ligne valide du 3 janvier pour l'exemple de requête serait une ligne dont saletime est `1/3/2008 00:00:00`.

Condition null

La condition null teste les valeurs null, lorsqu'une valeur est manquante ou inconnue.

Syntaxe

```
expression IS [ NOT ] NULL
```

Arguments

expression

N'importe quelle expression telle qu'une colonne.

IS NULL

Condition vraie lorsque la valeur de l'expression est null et fausse quand elle a une valeur.

IS NOT NULL

Condition fausse lorsque la valeur de l'expression est null et vraie quand elle a une valeur.

exemple

Cet exemple indique le nombre de fois où la table SALES contient null dans le champ QTYSOLD :

```
select count(*) from sales
where qtysold is null;
count
-------
0
(1 row)
```

Condition EXISTS

Les conditions EXIST testent l'existence de lignes dans une sous-requête et retournent la valeur true si la requête renvoie au moins une ligne. Si NOT n'est pas spécifié, la condition retourne true si une sous-requête ne renvoie aucune ligne.

Syntaxe

```
[ NOT ] EXISTS (table_subquery)
```

Arguments

EXISTS

Est vraie lorsque la table_subquery retourne au moins une ligne.

NOT EXISTS

Est vraie lorsque la table_subquery ne retourne pas de lignes.

table_subquery

Une sous-requête qui analyse une table avec une ou plusieurs colonnes et une ou plusieurs lignes.

exemple

Cet exemple retourne tous les identificateurs de date, une fois chacun, pour chaque date où une vente a eu lieu :

```
select dateid from date
where exists (
select 1 from sales
where date.dateid = sales.dateid
)
order by dateid;

dateid
--------
1827
1828
1829
...
```

Condition IN

Une condition IN teste une valeur d'adhésion dans une sous-requête ou un ensemble de valeurs.

Syntaxe

```
expression [ NOT ] IN (expr_list | table_subquery)
```

Arguments

expression

Une expression de type numeric, character ou datetime évaluée par rapport à expr_list ou table_subquery et qui doit être compatible avec le type de données de cette liste ou sous-requête.

expr_list

Une ou plusieurs expressions délimitées par des virgules, ou un ou plusieurs ensembles d'expressions délimitées par des virgules et entourées par des parenthèses.

table_subquery

Une sous-requête qui correspond à une table avec une ou plusieurs lignes, mais est limitée à une seule colonne dans la liste de sélection.

IN | NOT IN

IN retourne la valeur true si l'expression est membre de la liste d'expressions ou de la requête. NOT IN retourne la valeur true si l'expression n'est pas membre. IN et NOT IN retournent la valeur NULL et aucune ligne n'est retournée dans les cas suivants : si expression génère null, s'il n'y a aucune expr_list correspondante ou si les valeurs de table_subquery et au moins l'une de ces lignes de comparaison entraînent une valeur null.

Exemples

Les conditions suivantes sont vraies uniquement pour les valeurs répertoriées :

```
qtysold in (2, 4, 5)
date.day in ('Mon', 'Tues')
date.month not in ('Oct', 'Nov', 'Dec')
```

Optimisation pour les grandes listes IN

Afin d'optimiser les performances des requêtes, une liste IN qui inclut plus de 10 valeurs est analysée en interne comme un ensemble (array) scalaire. Les listes IN avec moins de 10 valeurs sont évaluées comme une série de prédicats OR. Cette optimisation est prise en charge pour les types de données SMALLINT, INTEGER, BIGINT, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE, TIMESTAMP et TIMESTAMPTZ.

Examinez la sortie EXPLAIN de la requête pour voir l'effet de cette optimisation. Par exemple :

```
explain select * from sales
QUERY PLAN
------------------------------------------------------------------------
XN Seq Scan on sales  (cost=0.00..6035.96 rows=86228 width=53)
Filter: (salesid = ANY ('{1,2,3,4,5,6,7,8,9,10,11}'::integer[]))
(2 rows)
```

Commandes SQL

Le langage SQL se compose des commandes que vous utilisez pour créer et manipuler des objets de base de données, exécuter des requêtes, charger des tables et modifier les données des tables.

Amazon Redshift est basé sur PostgreSQL. Amazon Redshift et PostgreSQL présentent un certain nombre de différences importantes dont vous devez être conscient lorsque vous concevez et

développez vos applications d'entrepôt des données. Pour plus d'informations sur les différences entre Amazon Redshift SQL et PostgreSQL, consultez Amazon Redshift et PostgreSQL.

> **Note**
>
> La taille maximale d'une instruction SQL est de 16 Mo.

Rubriques

- ABORT
- ALTER DATABASE
- ALTER DATASHARE
- ALTER DEFAULT PRIVILEGES
- MODIFIER LE SCHÉMA EXTERNE
- MODIFIER LA VUE EXTÉRIEURE
- ALTER FUNCTION
- ALTER GROUP
- ALTER IDENTITY PROVIDER
- MODIFIER LA POLITIQUE DE MASQUAGE
- ALTER MATERIALIZED VIEW
- ALTER RLS POLICY
- ALTER ROLE
- ALTER PROCEDURE
- ALTER SCHEMA
- ALTER SYSTEM
- ALTER TABLE
- ALTER TABLE APPEND
- ALTER USER
- ANALYSE
- ANALYZE COMPRESSION
- ATTACH MASKING POLICY

- ATTACH RLS POLICY
- BEGIN
- CALL
- ANNULER
- CLOSE
- COMMENT
- COMMIT
- COPY
- CREATE DATABASE
- CREATE DATASHARE
- CREATE EXTERNAL FUNCTION
- CRÉER UN MODÈLE EXTERNE
- CREATE EXTERNAL SCHEMA
- CREATE EXTERNAL TABLE
- CRÉER UNE VUE EXTERNE
- CREATE FUNCTION
- CREATE GROUP
- CREATE IDENTITY PROVIDER
- CREATE LIBRARY
- CREATE MASKING POLICY
- CREATE MATERIALIZED VIEW
- CREATE MODEL
- CREATE PROCEDURE
- CREATE RLS POLICY
- CREATE ROLE
- CREATE SCHEMA
- CREATE TABLE
- CREATE TABLE AS
- CREATE USER
- CREATE VIEW

- DEALLOCATE
- DECLARE
- DELETE
- DESC DATASHARE
- DESC IDENTITY PROVIDER
- DETACH MASKING POLICY
- DETACH RLS POLICY
- DROP DATABASE
- DROP DATASHARE
- SUPPRIMER LA VUE EXTÉRIEURE
- DROP FUNCTION
- DROP GROUP
- DROP IDENTITY PROVIDER
- DROP LIBRARY
- DROP MASKING POLICY
- DROP MODEL
- DROP MATERIALIZED VIEW
- DROP PROCEDURE
- DROP RLS POLICY
- DROP ROLE
- DROP SCHEMA
- DROP TABLE
- DROP USER
- DROP VIEW
- FIN
- EXECUTE
- EXPLAIN
- FETCH
- GRANT
- INSERT

- INSERT (table externe)
- LOCK
- MERGE
- PREPARE
- REFRESH MATERIALIZED VIEW
- RESET
- REVOKE
- ROLLBACK
- SELECT
- SELECT INTO
- SET
- SET SESSION AUTHORIZATION
- SET SESSION CHARACTERISTICS
- MONTRER
- SHOW COLUMNS
- SHOW EXTERNAL TABLE
- SHOW DATABASES
- AFFICHER LES SUBVENTIONS
- SHOW MODEL
- SHOW DATASHARES
- SHOW PROCEDURE
- SHOW SCHEMAS
- SHOW TABLE
- SHOW TABLES
- SHOW VIEW
- START TRANSACTION
- TRUNCATE
- UNLOAD
- UPDATE
- USE

- VACUUM

ABORT

Arrête la transaction en cours d'exécution et ignore toutes les mises à jour effectuées par cette transaction. ABORT n'a aucun effet sur les transactions déjà terminées.

Cette commande effectue la même fonction que la commande ROLLBACK. Pour obtenir des informations, consultez ROLLBACK.

Syntaxe

```
ABORT [ WORK | TRANSACTION ]
```

Paramètres

WORK

Mot-clé facultatif.

TRANSACTION

Mot-clé facultatif ; WORK et TRANSACTION sont synonymes.

exemple

L'exemple suivant crée une table, puis démarre une transaction où les données sont insérées dans la table. La commande ABORT annule ensuite l'insertion des données pour laisser la table vide.

La commande suivante crée un exemple de table appelée MOVIE_GROSS :

```
create table movie_gross( name varchar(30), gross bigint );
```

La prochaine série de commandes démarre une transaction qui insère deux lignes de données dans la table :

```
begin;

insert into movie_gross values ( 'Raiders of the Lost Ark', 23400000);
```

```
insert into movie_gross values ( 'Star Wars', 10000000 );
```

Puis, la commande suivante sélectionne les données de la table pour montrer qu'elles ont été correctement insérées :

```
select * from movie_gross;
```

La sortie de la commande montre que les deux lignes ont été insérées avec succès :

```
         name           |  gross
------------------------+----------
Raiders of the Lost Ark | 23400000
Star Wars               | 10000000
(2 rows)
```

Cette commande restaure les modifications des données à l'emplacement où la transaction a commencé :

```
abort;
```

La sélection de données de la table affiche maintenant une table vide :

```
select * from movie_gross;

 name | gross
------+-------
(0 rows)
```

ALTER DATABASE

Modifie les attributs d'une base de données.

Privilèges requis

Pour utiliser ALTER DATABASE, l'un des privilèges suivants est requis.

- Superuser
- Utilisateurs disposant du privilège ALTER DATABASE

- Propriétaire de la base de données

Syntaxe

```
ALTER DATABASE database_name
{
  RENAME TO new_name
  | OWNER TO new_owner
  | [ CONNECTION LIMIT { limit | UNLIMITED } ]
    [ COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE } ]
    [ ISOLATION LEVEL { SERIALIZABLE | SNAPSHOT } ]
| INTEGRATION
 {
  REFRESH { { ALL | INERROR } TABLES [ IN SCHEMA schema [, ...] ] | TABLE schema.table
 [, ...] }
   | SET
     [ QUERY_ALL_STATES [=] { TRUE | FALSE } ]
     [ ACCEPTINVCHARS [=] { TRUE | FALSE } ]
     [ REFRESH_INTERVAL <interval> ]
     [ TRUNCATECOLUMNS [=] { TRUE | FALSE } ]
     [ HISTORY_MODE [=] {TRUE | FALSE} [ FOR { {ALL} TABLES [IN SCHEMA schema [, ...] ]
 | TABLE schema.table [, ...] } ] ]
 }
}
```

Paramètres

database_name

Nom de la base de données à modifier. Généralement, vous modifiez une base de données à laquelle vous n'êtes pas connecté ; quoi qu'il en soit, les modifications ne prennent effet que lors des séances suivantes. Vous pouvez modifier le propriétaire de la base de données, mais vous ne pouvez pas le renommer :

```
alter database tickit rename to newtickit;
ERROR:  current database may not be renamed
```

RENAME TO

Renomme la base de données spécifiée. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs. Vous ne pouvez pas renommer les bases de données dev, padb_harvest,

template0, template1 ou sys:internal, et vous ne pouvez pas renommer la base de données active. Seul le propriétaire de la base de données ou un superuser (p. 999) peut renommer une base de données ; les propriétaires qui ne sont pas des super-utilisateurs doivent aussi avoir le privilège CREATEDB.

nouveau_nom

Nouveau nom de la base de données.

OWNER TO

Modifie le propriétaire de la base de données spécifiée. Vous pouvez modifier le propriétaire de la base de données active ou d'une autre base de données. Seul un super-utilisateur peut changer le propriétaire.

nouveau_propriétaire

Nouveau propriétaire de la base de données. Le nouveau propriétaire doit être un utilisateur de base de données existant avec des privilèges en écriture. Pour plus d'informations sur les privilèges d'utilisateur, consultez GRANT.

CONNECTION LIMIT { limite | UNLIMITED }

Le nombre maximum de connexions à la base de données que les utilisateurs sont autorisés à ouvrir simultanément. La limite n'est pas appliquée pour les super-utilisateurs. Utilisez le mot-clé UNLIMITED pour autoriser le nombre maximum de connexions simultanées. Une limite sur le nombre de connexions pour chaque utilisateur peut également s'appliquer. Pour plus d'informations, consultez CREATE USER. La valeur par défaut est UNLIMITED. Pour afficher les connexions en cours, interrogez la vue système STV_SESSIONS.

> **Note**
>
> Si les deux limites de connexion (utilisateurs et base de données) s'appliquent, un emplacement de connexion inutilisé situé entre les deux limites doit également être disponible lorsqu'un utilisateur tente de se connecter.

COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE }

Clause spécifiant si la recherche de chaînes ou la comparaison est sensible à la casse ou non.

Vous pouvez modifier la sensibilité à la casse de la base de données active qui est vide.

Vous devez disposer du privilège sur la base de données active pour modifier la sensibilité à la casse. Les super-utilisateurs ou propriétaires de base de données disposant du privilège CREATE DATABASE peuvent également modifier la sensibilité à la casse de la base de données.

ISOLATION LEVEL { SERIALIZABLE | SNAPSHOT }

Clause qui spécifie le niveau d'isolation utilisé lorsque les requêtes sont exécutées sur une base de données.

- Isolation SERIALIZABLE : offre une mise en série complète pour les transactions simultanées. Pour de plus amples informations, veuillez consulter Isolement sérialisable.
- Isolation SNAPSHOT : offre un niveau d'isolation avec protection contre les conflits de mise à jour et de suppression

Pour plus d'informations sur les niveaux d'isolation, consultez CREATE DATABASE.

Tenez compte des éléments suivants lorsque vous modifiez le niveau d'isolation d'une base de données :

- Vous devez disposer du privilège super-utilisateur ou CREATE DATABASE sur la base de données active pour modifier le niveau d'isolation de la base de données.
- Vous ne pouvez pas modifier le niveau d'isolation de la base de données `dev`.
- Vous ne pouvez pas modifier le niveau d'isolation au sein d'un bloc de transaction.
- La commande de modification du niveau d'isolation échoue si d'autres utilisateurs sont connectés à la base de données.
- La commande de modification du niveau d'isolation peut modifier les paramètres de niveau d'isolation de la séance en cours.

INTÉGRATION

Modifiez une base de données d'intégration zéro ETL.

ACTUALISER {{ALL | INERROR} TABLES [DANS LE SCHÉMA DU SCHÉMA [,...]] | TABLEAU schema.table [,...]}

Clause spécifiant si Amazon Redshift actualisera toutes les tables ou les tables présentant des erreurs dans le schéma ou la table spécifiés. L'actualisation déclenchera la réplication complète des tables du schéma ou de la table spécifiés à partir de la base de données source.

Pour plus d'informations, consultez la section Intégrations Zero-ETL dans le guide de gestion Amazon Redshift. Pour plus d'informations sur les statuts d'intégration, consultez SVV_INTEGRATION_TABLE_STATE et SVV_INTEGRATION.

QUERY_ALL_STATES [=] {VRAI | FAUX}

La clause QUERY_ALL_STATES indique si les tables d'intégration zéro ETL peuvent être interrogées dans tous les états (,, et). `Synced` `Failed` `ResyncRequired` `ResyncInitiated` Par défaut, une table d'intégration zéro ETL ne peut être interrogée que dans son état. `Synced`

ACCEPTINVCHARS [=] {VRAI | FAUX}

La clause ACCEPTINVCHARS indique si les tables d'intégration zéro ETL continuent à être ingérées lorsque des caractères non valides sont détectés pour le type de données VARCHAR. Lorsque des caractères non valides sont détectés, ils sont remplacés par un ? caractère par défaut.

INTERVALLE DE RAFRAÎCHISSEMENT <interval>

La clause REFRESH_INTERVAL définit l'intervalle de temps approximatif, en secondes, pour actualiser les données de la source zéro ETL vers la base de données cible. La valeur `interval` peut être définie entre 0 et 432 000 secondes (5 jours) pour les intégrations sans ETL dont le type de source est Aurora MySQL, Aurora PostgreSQL ou RDS for MySQL. Pour les intégrations Amazon DynamoDB Zero-ETL, les valeurs peuvent être définies entre 900 et `interval` 432 000 secondes (15 minutes et 5 jours).

Pour plus d'informations sur la création de bases de données avec des intégrations sans ETL, consultez la section Création de bases de données de destination dans Amazon Redshift dans le guide de gestion Amazon Redshift.

TRONCATCOLUMNS [=] {VRAI | FAUX}

La clause TRUNCATECOLUMNS indique si les tables d'intégration Zero-ETL continuent à être ingérées lorsque les valeurs des attributs de colonne VARCHAR ou SUPER dépassent les limites. Lorsque `TRUE` les valeurs sont tronquées pour tenir dans la colonne et que les valeurs des attributs JSON débordants sont tronquées pour tenir dans la colonne SUPER.

HISTORY_MODE [=] {TRUE | FALSE} [POUR {{TOUTES} TABLES [DANS LE SCHÉMA [,...]] | TABLE schema.table [,...]}]

Clause qui indique si Amazon Redshift définira le mode historique pour toutes les tables ou les tables du schéma spécifié qui participent à l'intégration zéro ETL. Cette option s'applique uniquement aux bases de données créées pour une intégration zéro ETL.

La clause HISTORY_MODE peut être définie sur ou. `TRUE` `FALSE` L'argument par défaut est `FALSE`. L'activation ou la désactivation du mode historique ne s'applique qu'aux tables qui sont

dans `Synced` cet état. Pour plus d'informations sur HISTORY_MODE, consultez la section Mode historique dans le guide de gestion Amazon Redshift.

Notes d’utilisation

Les commandes ALTER DATABASE s’appliquent aux séances ultérieures, pas aux séances en cours. Vous devez vous reconnecter à la base de données modifiée pour voir l’effet de la modification.

Exemples

L’exemple suivant renomme une base de données nommée TICKIT_SANDBOX en TICKIT_TEST :

```
alter database tickit_sandbox rename to tickit_test;
```

L’exemple suivant modifie le propriétaire de la base de données TICKIT (la base de données active) en DWUSER :

```
alter database tickit owner to dwuser;
```

L’exemple suivant modifie la sensibilité à la casse de la base de données sampledb :

```
ALTER DATABASE sampledb COLLATE CASE_INSENSITIVE;
```

L’exemple suivant montre comment modifier une base de données nommée **`sampledb`** avec un niveau d’isolation SNAPSHOT.

```
ALTER DATABASE sampledb ISOLATION LEVEL SNAPSHOT;
```

L’exemple suivant actualise les tables **`schema1.sample_table1`** et **`schema2.sample_table2`** dans la base de données **`sample_integration_db`** dans le cadre de votre intégration zéro ETL.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH TABLE schema1.sample_table1,
 schema2.sample_table2;
```

L’exemple suivant actualise toutes les tables synchronisées et défaillantes au sein de votre intégration zéro ETL.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH ALL tables;
```

L'exemple suivant définit l'intervalle d'actualisation pour les intégrations sans ETL à 600 secondes.

```
ALTER DATABASE sample_integration_db INTEGRATION SET REFRESH_INTERVAL 600;
```

L'exemple suivant actualise toutes les tables figurant `ErrorState` dans le schéma**`sample_schema`**.

```
ALTER DATABASE sample_integration_db INTEGRATION REFRESH INERROR TABLES in SCHEMA
 sample_schema;
```

L'exemple suivant active le mode historique pour le tableau`myschema.table1`.

```
ALTER DATABASE sample_integration_db INTEGRATION SET HISTORY_MODE = true FOR TABLE
 myschema.table1
```

L'exemple suivant active le mode historique pour toutes les tables de`myschema`.

```
ALTER DATABASE sample_integration_db INTEGRATION SET HISTORY_MODE = true for ALL TABLES
 IN SCHEMA myschema
```

ALTER DATASHARE

Modifie la définition d’une unité de partage des données. Vous pouvez ajouter ou supprimer des objets à l’aide de la commande ALTER DATASHARE. Vous ne pouvez modifier une unité de partage des données que dans la base de données actuelle. Ajoutez ou supprimez des objets de la base de données associée à une unité de partage des données. Le propriétaire de l’unité de partage des données disposant des autorisations requises sur les objets d’unité de partage des données à ajouter ou à supprimer peut modifier l’unité de partage des données.

Privilèges requis

Les privilèges suivants sont requis pour ALTER DATASHARE :

- Super-utilisateur.
- Utilisateur disposant du privilège ALTER DATASHARE.
- Utilisateurs disposant du privilège ALTER ou ALL sur l’unité de partage des données.

- Pour ajouter des objets spécifiques à une unité de partage des données, les utilisateurs doivent avoir le privilège sur les objets. Dans ce cas, les utilisateurs doivent être les propriétaires des objets ou disposer des privilèges SELECT, USAGE ou ALL sur les objets.

Syntaxe

La syntaxe suivante illustre comment ajouter ou supprimer des objets à l'unité de partage des données.

```
ALTER DATASHARE datashare_name { ADD | REMOVE } {
TABLE schema.table [, ...]
| SCHEMA schema [, ...]
| FUNCTION schema.sql_udf (argtype,...) [, ...]
| ALL TABLES IN SCHEMA schema [, ...]
| ALL FUNCTIONS IN SCHEMA schema [, ...] }
```

La syntaxe suivante montre la manière de configurer les propriétés de l'unité de partage des données.

```
ALTER DATASHARE datashare_name {
[ SET PUBLICACCESSIBLE [=] TRUE | FALSE ]
[ SET INCLUDENEW [=] TRUE | FALSE FOR SCHEMA schema ] }
```

Paramètres

datashare_name

Nom de l'unité de partage des données à modifier.

ADD | REMOVE

Clause spécifiant s'il faut ajouter ou supprimer des objets de l'unité de partage des données.

TABLE schema.table [, ...]

Nom de la table ou de la vue dans le schéma spécifié à ajouter à l'unité de partage des données.

SCHEMA schema [, ...]

Nom du schéma à ajouter à l'unité de partage des données.

FUNCTION schema.sql_udf (argtype,...) [, ...]

Nom de la fonction SQL définie par l'utilisateur avec des types d'arguments à ajouter à l'unité de partage des données.

ALL TABLES IN SCHEMA schéma [, ...]

Clause spécifiant s'il faut ajouter toutes les tables et vues du schéma spécifié à l'unité de partage des données.

ALL FUNCTIONS IN SCHEMA schema [, ...] }

Clause spécifiant l'ajout de toutes les fonctions du schéma spécifié à l'unité de partage des données.

[SET PUBLICACCESSIBLE [=] TRUE | FALSE]

Clause spécifiant si une unité de partage des données peut être partagée avec des clusters accessibles au public.

[SET INCLUDENEW [=] TRUE | FALSE FOR SCHEMA schema]

Clause qui indique s'il faut ajouter au partage de données les futures tables, vues ou fonctions SQL définies par l'utilisateur (UDFs) créées dans le schéma spécifié. Les tables, les vues ou le code SQL UDFs actuels du schéma spécifié ne sont pas ajoutés au partage de données. Seuls les super-utilisateurs peuvent modifier cette propriété pour chaque paire de datashare-schéma. Par défaut, la clause INCLUDENEW est false.

Notes d'utilisation d'ALTER DATASHARE

- Les utilisateurs suivants peuvent modifier une unité de partage des données :
  - Super-utilisateur
  - Le propriétaire de l'unité de partage des données
  - Les utilisateurs disposant du privilège ALTER ou ALL sur l'unité de partage des données
- Pour ajouter des objets spécifiques à une unité de partage des données, ces utilisateurs doivent avoir le privilège adéquat sur les objets. Les utilisateurs doivent être les propriétaires des objets ou avoir les privilèges SELECT, USAGE ou ALL sur les objets.
- Vous pouvez partager des schémas, des tables, des vues standard, des vues à liaison tardive, des vues matérialisées et des fonctions SQL définies par l'utilisateur (). UDFs Ajoutez d'abord un schéma à une unité de partage des données avant d'ajouter d'autres objets dans le schéma.

Lorsque vous ajoutez un schéma, Amazon Redshift n'ajoute pas tous les objets qu'il contient. Vous devez les ajouter explicitement.

- Nous vous recommandons de créer des AWS Data Exchange partages de données avec le paramètre accessible au public activé.
- En général, nous vous recommandons de ne pas modifier un partage de AWS Data Exchange données pour désactiver l'accessibilité publique à l'aide de l'instruction ALTER DATASHARE. Si vous le faites, les Comptes AWS qui ont accès à l'unité de partage des données perdent l'accès si leurs clusters sont accessibles au public. L'exécution de ce type de modification peut enfreindre les termes des produits de données dans AWS Data Exchange. Pour une exception à cette recommandation, reportez-vous à ce qui suit.

  L'exemple suivant montre une erreur lorsqu'un partage de AWS Data Exchange données est créé alors que le paramètre est désactivé.

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
ERROR:  Alter of ADX-managed datashare salesshare requires session variable
 datashare_break_glass_session_var to be set to value 'c670ba4db22f4b'
```

  Pour autoriser la modification d'un AWS Data Exchange partage de données afin de désactiver le paramètre accessible au public, définissez la variable suivante et réexécutez l'instruction ALTER DATASHARE.

```
SET datashare_break_glass_session_var to 'c670ba4db22f4b';
```

```
ALTER DATASHARE salesshare SET PUBLICACCESSIBLE FALSE;
```

  Dans ce cas, Amazon Redshift génère une valeur aléatoire à usage unique pour définir la variable de séance afin d'autoriser ALTER DATASHARE SET PUBLICACCESSIBLE FALSE pour une unité de partage des données AWS Data Exchange .

Exemples

L'exemple suivant ajoute le `public` schéma au partage de données. `salesshare`

```
ALTER DATASHARE salesshare ADD SCHEMA public;
```

L'exemple suivant ajoute la table `public.tickit_sales_redshift` à l'unité de partage des données `salesshare`.

```
ALTER DATASHARE salesshare ADD TABLE public.tickit_sales_redshift;
```

L'exemple suivant ajoute toutes les tables à l'unité de partage des données `salesshare`.

```
ALTER DATASHARE salesshare ADD ALL TABLES IN SCHEMA PUBLIC;
```

L'exemple suivant supprime la table `public.tickit_sales_redshift` de l'exemple de l'unité de partage des données `salesshare`.

```
ALTER DATASHARE salesshare REMOVE TABLE public.tickit_sales_redshift;
```

ALTER DEFAULT PRIVILEGES

Définit l'ensemble d'autorisations d'accès par défaut à appliquer aux objets créés à l'avenir par l'utilisateur spécifié. Par défaut, les utilisateurs peuvent ne modifier que leurs propres autorisations d'accès par défaut. Seul un superutilisateur peut définir des autorisations par défaut pour les autres utilisateurs.

Vous pouvez appliquer les privilèges par défaut à des rôles, des utilisateurs ou des groupes d'utilisateurs. Vous pouvez définir des autorisations par défaut globalement pour tous les objets créés dans la base de données actuelle, ou pour les objets créés uniquement dans les schémas spécifiés.

Les autorisations par défaut s'appliquent uniquement aux nouveaux objets. L'exécution de ALTER DEFAULT PRIVILEGES ne modifie pas les autorisations sur les objets existants. Pour accorder des autorisations sur tous les objets actuels et futurs créés par n'importe quel utilisateur au sein d'une base de données ou d'un schéma, voir Autorisations étendues.

Pour afficher les informations sur les privilèges par défaut des utilisateurs de base de données, interrogez la table catalogue système PG_DEFAULT_ACL.

Pour plus d'informations sur les privilèges, consultez GRANT.

Privilèges requis

Les privilèges suivants sont requis pour ALTER DEFAULT PRIVILEGES :

- Superuser
- Utilisateurs disposant du privilège ALTER DEFAULT PRIVILEGES
- Utilisateurs modifiant leurs propres privilèges d'accès par défaut
- Utilisateurs définissant des privilèges pour les schémas pour lesquels ils disposent de privilèges d'accès

Syntaxe

```
ALTER DEFAULT PRIVILEGES
    [ FOR USER target_user [, ...] ]
    [ IN SCHEMA schema_name [, ...] ]
    grant_or_revoke_clause

where grant_or_revoke_clause is one of:

GRANT { { SELECT | INSERT | UPDATE | DELETE | DROP | REFERENCES | TRUNCATE } [,...] |
 ALL [ PRIVILEGES ] }
 ON TABLES
 TO { user_name [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
 ON FUNCTIONS
 TO { user_name [ WITH GRANT OPTION ] |  ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
 ON PROCEDURES
 TO { user_name [ WITH GRANT OPTION ] |  ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

REVOKE [ GRANT OPTION FOR ] { { SELECT | INSERT | UPDATE | DELETE | REFERENCES |
 TRUNCATE } [,...] | ALL [ PRIVILEGES ] }
 ON TABLES
 FROM user_name [, ...] [ RESTRICT ]

REVOKE  { { SELECT | INSERT | UPDATE | DELETE | REFERENCES | TRUNCATE } [,...] | ALL
 [ PRIVILEGES ] }
 ON TABLES
 FROM { ROLE role_name | GROUP group_name | PUBLIC } [, ...] [ RESTRICT ]
```

```
REVOKE [ GRANT OPTION FOR ] { EXECUTE | ALL [ PRIVILEGES ] }
 ON FUNCTIONS
 FROM user_name [, ...] [ RESTRICT ]

REVOKE { EXECUTE | ALL [ PRIVILEGES ] }
 ON FUNCTIONS
 FROM { ROLE role_name | GROUP group_name | PUBLIC } [, ...] [ RESTRICT ]

REVOKE [ GRANT OPTION FOR ] { EXECUTE | ALL [ PRIVILEGES ] }
 ON PROCEDURES
 FROM user_name [, ...] [ RESTRICT ]

REVOKE { EXECUTE | ALL [ PRIVILEGES ] }
 ON PROCEDURES
 FROM { ROLE role_name | GROUP group_name | PUBLIC } [, ...] [ RESTRICT ]
```

Paramètres

FOR USER utilisateur_cible

Facultatif. Nom de l'utilisateur pour lequel les privilèges par défaut sont définis. Seul un super-utilisateur peut spécifier les privilèges par défaut d'autres utilisateurs. La valeur par défaut est l'utilisateur actif.

IN SCHEMA nom_schéma

Facultatif. Si une clause IN SCHEMA s'affiche, les privilèges par défaut spécifiés s'appliquent aux nouveaux objets créés dans le nom_schéma spécifié. Dans ce cas, l'utilisateur ou groupe d'utilisateurs qui est la cible d'ALTER DEFAULT PRIVILEGES doit avoir le privilège CREATE pour le schéma spécifié. Les privilèges par défaut qui sont spécifiques à un schéma sont ajoutés aux privilèges par défaut globaux existants. Par défaut, les privilèges par défaut sont appliqués globalement à l'ensemble de la base de données.

GRANT

Ensemble de privilèges à accorder aux utilisateurs ou aux groupes spécifiés pour toutes les nouvelles tables et vues, fonctions ou procédures stockées créées par l'utilisateur spécifié. Vous pouvez définir les mêmes privilèges et options avec la clause GRANT qu'avec la commande GRANT.

WITH GRANT OPTION

Clause indiquant que l'utilisateur qui reçoit les privilèges peut à son tour accorder les mêmes privilèges à d'autres personnes. Vous ne pouvez pas accorder WITH GRANT OPTION à un groupe ou à PUBLIC.

TO user_name | ROLE role_name | GROUP group_name

Nom de l'utilisateur, du rôle ou du groupe d'utilisateurs auquel les privilèges par défaut spécifiés s'appliquent.

REVOKE

Ensemble des privilèges à retirer aux utilisateurs ou groupes spécifiés pour toutes les nouvelles tables, fonctions ou procédures stockées créées par l'utilisateur spécifié. Vous pouvez définir les mêmes privilèges et options avec la clause REVOKE qu'avec la commande REVOKE.

GRANT OPTION FOR

Clause qui retire uniquement la possibilité d'accorder un privilège spécifié à d'autres utilisateurs et qui ne retire pas le privilège lui-même. Vous ne pouvez pas retirer GRANT OPTION d'un groupe ou de PUBLIC.

FROM user_name | ROLE role_name | GROUP group_name

Le nom de l'utilisateur, du rôle ou du groupe d'utilisateurs à partir duquel les privilèges spécifiés sont révoqués par défaut.

RESTRICT

L'option RESTRICT révoque uniquement les privilèges que l'utilisateur a directement accordés. Il s'agit de l'option par défaut.

Exemples

Supposons que vous souhaitiez autoriser n'importe quel utilisateur du groupe d'utilisateurs `report_readers` à consulter toutes les tables et vues créées par l'utilisateur`report_admin`. Dans ce cas, exécutez la commande suivante en tant que super-utilisateur.

```
alter default privileges for user report_admin grant select on tables to group
 report_readers;
```

Dans l'exemple suivant, la première commande accorde des privilèges SELECT sur toutes les nouvelles tables que vous créez. Chaque fois que vous créez une nouvelle vue, vous devez

explicitement accorder des privilèges à la vue ou réexécuter la `alter default privileges` commande.

```
alter default privileges grant select on tables to public;
```

L'exemple suivant accorde le privilège INSERT au groupe d'utilisateurs `sales_admin` pour toutes les nouvelles tables et vues que vous créez dans le schéma `sales`.

```
alter default privileges in schema sales grant insert on tables to group sales_admin;
```

L'exemple suivant inverse la commande ALTER DEFAULT PRIVILEGES de l'exemple précédent.

```
alter default privileges in schema sales revoke insert on tables from group
 sales_admin;
```

Par défaut, le groupe d'utilisateurs PUBLIC a l'autorisation d'exécution pour toutes les nouvelles fonctions définies par l'utilisateur. Pour retirer les autorisations d'exécution `public` de vos nouvelles fonctions, puis n'accorder l'autorisation d'exécution qu'au groupe d'utilisateurs `dev_test`, exécutez les commandes suivantes.

```
alter default privileges revoke execute on functions from public;
alter default privileges grant execute on functions to group dev_test;
```

MODIFIER LE SCHÉMA EXTERNE

Modifie un schéma externe existant dans la base de données actuelle. Seuls les propriétaires du schéma, les super-utilisateurs ou les utilisateurs disposant des privilèges ALTER sur le schéma peuvent le modifier. Seuls les schémas externes créés à partir de DATA CATALOG, KAFKA ou MSK peuvent être modifiés.

Le propriétaire de ce schéma est l'auteur de la commande CREATE EXTERNAL SCHEMA. Pour transférer la propriété d'un schéma externe, utilisez ALTER SCHEMA pour modifier le propriétaire. Pour accorder l'accès au schéma à d'autres utilisateurs ou groupes d'utilisateurs, utilisez la commande GRANT.

Vous ne pouvez pas utiliser les commandes GRANT ou REVOKE pour des autorisations concernant une table externe. Vous pouvez en revanche accorder ou révoquer les autorisations pour le schéma externe.

Pour plus d'informations, consultez les ressources suivantes :

- ALTER SCHEMA
- GRANT
- REVOKE
- CREATE EXTERNAL SCHEMA
- Activation de l'authentification mTLS pour un schéma externe existant

Pour afficher les détails des schémas externes, interrogez la vue système SVV_EXTERNAL_SCHEMAS. Pour de plus amples informations, veuillez consulter SVV_EXTERNAL_SCHEMAS.

Syntaxe

```
ALTER EXTERNAL SCHEMA schema_name
[ IAM_ROLE [ default | 'SESSION' | 'arn:aws:iam::<AWS account-id>:role/<role-name>' ] ]
[ AUTHENTICATION [ none | iam | mtls] ]
[ AUTHENTICATION_ARN 'acm-certificate-arn' | SECRET_ARN 'asm-secret-arn' ]
[ URI 'Kafka bootstrap URL' ]
```

Si vous utilisez un schéma externe pour l'ingestion du streaming et que vous souhaitez implémenter le protocole TLS mutuel pour l'authentification, vous pouvez exécuter une commande telle que la suivante, qui spécifie l'authentification mTLS et l'ARN du certificat ACM dans ACM.

```
ALTER EXTERNAL SCHEMA schema_name
AUTHENTICATION mtls
AUTHENTICATION_ARN 'arn:aws:acm:us-east-1:444455556666:certificate/certificate_ID';
```

Vous pouvez également modifier l'authentification mTLS, en référence à l'ARN secret dans Secrets Manager.

```
ALTER EXTERNAL SCHEMA schema_name
AUTHENTICATION mtls
SECRET_ARN 'arn:aws:secretsmanager:us-east-1:012345678910:secret:myMTLSSecret';
```

L'exemple suivant montre comment modifier l'URI pour ALTER EXTERNAL SCHEMA :

```
ALTER EXTERNAL SCHEMA schema_name
```

```
URI 'lkc-ghidef-67890.centralus.azure.glb.confluent.cloud:9092';
```

L'exemple suivant montre comment modifier le rôle IAM pour ALTER EXTERNAL SCHEMA :

```
ALTER EXTERNAL SCHEMA schema_name
IAM_ROLE 'arn:aws:iam::012345678901:role/testrole';
```

Paramètres

<role-name>IAM_ROLE [par défaut | 'SESSION' | 'arn:aws:iam : :< account-id>:role/ ']AWS

Utilisez le `default` mot clé pour qu'Amazon Redshift utilise le rôle IAM défini par défaut.

Utilisez `'SESSION'` si vous vous connectez à votre cluster Amazon Redshift à l'aide d'une identité fédérée et que vous accédez aux tables à partir du schéma externe créé à l'aide de cette commande.

Voir CRÉER UN SCHÉMA EXTERNE pour plus d'informations.

AUTHENTICATION

Type d'authentification défini pour l'ingestion en streaming. L'ingestion du streaming avec des types d'authentification fonctionne avec Apache Kafka, Confluent Cloud et Amazon Managed Streaming for Apache Kafka. Voir CRÉER UN SCHÉMA EXTERNE pour plus d'informations.

AUTHENTIFICATION_ARN

L'ARN du AWS Certificate Manager certificat utilisé par Amazon Redshift pour l'authentification MTLS avec Apache Kafka, Confluent Cloud ou Amazon Managed Streaming for Apache Kafka (Amazon MSK). L'ARN est disponible dans la console ACM lorsque vous choisissez le certificat émis.

SECRET ARN

Le nom de ressource Amazon (ARN) d'un secret pris en charge créé à l'aide de AWS Secrets Manager. Pour plus d'informations sur la création et la récupération d'un ARN pour un secret, consultez les sections Gérer les secrets AWS Secrets Manager dans le guide de AWS Secrets Manager l'utilisateur et Extraction du nom de ressource Amazon (ARN) du secret dans Amazon Redshift.

URI

URL de démarrage du cluster Apache Kafka, Confluent Cloud ou Amazon Managed Streaming for Apache Kafka (Amazon MSK). Le point de terminaison doit être accessible (routable) à partir du cluster Amazon Redshift. Voir CRÉER UN SCHÉMA EXTERNE pour plus d'informations.

MODIFIER LA VUE EXTÉRIEURE

Utilisez la commande ALTER EXTERNAL VIEW pour mettre à jour votre vue externe. Selon les paramètres que vous utilisez, d'autres moteurs SQL comme Amazon Athena et Amazon EMR Spark, qui peuvent également référencer cette vue, peuvent être affectés. Pour plus d'informations sur les vues du catalogue de données, consultez la section AWS Glue Data Catalog vues.

Syntaxe

```
ALTER EXTERNAL VIEW schema_name.view_name
{catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name |
 external_schema_name.view_name}
[FORCE] { AS (query_definition) | REMOVE DEFINITION }
```

Paramètres

schema_name.view_name

Le schéma attaché à votre AWS Glue base de données, suivi du nom de la vue.

catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name | external_schema_name.view_name

Notation du schéma à utiliser lors d'une modification de la vue. Vous pouvez spécifier d'utiliser une base de AWS Glue Data Catalog données Glue que vous avez créée ou un schéma externe que vous avez créé. Consultez CREATE DATABASE et CREATE EXTERNAL SCHEMA pour plus d'informations.

FORCE

Si la définition de la vue AWS Lake Formation doit être mise à jour même si les objets référencés dans le tableau sont incompatibles avec les autres moteurs SQL. Si Lake Formation met à jour la vue, celle-ci est considérée comme obsolète pour les autres moteurs SQL jusqu'à ce que ces moteurs soient également mis à jour.

AS query_definition

Définition de la requête SQL exécutée par Amazon Redshift pour modifier la vue.

REMOVE DEFINITION

S'il faut supprimer et recréer les vues. Les vues doivent être supprimées et recréées pour les marquer comme PROTECTED.

Exemples

L'exemple suivant modifie une vue du catalogue de données nommée sample_schema.glue_data_catalog_view.

```
ALTER EXTERNAL VIEW sample_schema.glue_data_catalog_view
FORCE
REMOVE DEFINITION
```

ALTER FUNCTION

Renomme une fonction ou change de propriétaire. Le nom de la fonction et les types de données sont obligatoires. Seul le propriétaire ou un superutilisateur peut renommer une fonction. Seul un superutilisateur peut modifier le propriétaire d'une fonction.

Syntaxe

```
ALTER FUNCTION function_name ( { [ py_arg_name py_arg_data_type | sql_arg_data_type }
 [ , ... ] ] )
    RENAME TO new_name
```

```
ALTER FUNCTION function_name ( { [ py_arg_name py_arg_data_type | sql_arg_data_type }
 [ , ... ] ] )
    OWNER TO { new_owner | CURRENT_USER | SESSION_USER }
```

Paramètres

function_name

Nom de la fonction à modifier. Spécifiez le nom de la fonction dans le chemin de recherche actuel ou utilisez le format schema_name.function_name pour utiliser un schéma spécifique.

py_arg_name py_arg_data_type | sql_arg_data_type

Facultatif. Une liste de noms d'arguments d'entrée et de types de données pour la fonction définie par l'utilisateur Python, ou une liste de types de données d'arguments d'entrée pour la fonction SQL définie par l'utilisateur.

nouveau_nom

Nouveau nom pour la fonction définie par l'utilisateur.

new_owner | CURRENT_USER | SESSION_USER

Un nouveau propriétaire pour la fonction définie par l'utilisateur.

Exemples

L'exemple suivant change le nom d'une fonction de `first_quarter_revenue` à`quarterly_revenue`.

```
ALTER FUNCTION first_quarter_revenue(bigint, numeric, int)
         RENAME TO quarterly_revenue;
```

L'exemple suivant remplace le propriétaire de la `quarterly_revenue` fonction par`etl_user`.

```
ALTER FUNCTION quarterly_revenue(bigint, numeric) OWNER TO etl_user;
```

ALTER GROUP

Modifie un groupe d’utilisateurs. Utilisez cette commande pour ajouter des utilisateurs au groupe, en supprimer ou renommer le groupe.

Syntaxe

```
ALTER GROUP group_name
{
ADD USER username [, ... ] |
DROP USER username [, ... ] |
RENAME TO new_name
}
```

Paramètres

nom_groupe

Nom du groupe d'utilisateurs à modifier.

ADD

Ajoute un utilisateur à un groupe d'utilisateurs.

DROP

Supprime un utilisateur d'un groupe d'utilisateurs.

nom d'utilisateur

Nom de l'utilisateur à ajouter au groupe ou à supprimer du groupe.

RENAME TO

Renomme le groupe d'utilisateurs. Les noms de groupe commençant par deux tirets bas sont réservés pour un usage Amazon Redshift interne. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

nouveau_nom

Nouveau nom du groupe d'utilisateurs.

Exemples

L'exemple suivant ajoute un utilisateur nommé DWUSER au groupe ADMIN_GROUP.

```
ALTER GROUP admin_group
ADD USER dwuser;
```

L'exemple suivant renomme le groupe ADMIN_GROUP en ADMINISTRATORS.

```
ALTER GROUP admin_group
RENAME TO administrators;
```

L'exemple suivant ajoute deux utilisateurs au groupe ADMIN_GROUP.

```
ALTER GROUP admin_group
```

```
ADD USER u1, u2;
```

L'exemple suivant supprime deux utilisateurs du groupe ADMIN_GROUP.

```
ALTER GROUP admin_group
DROP USER u1, u2;
```

ALTER IDENTITY PROVIDER

Modifie un fournisseur d'identité pour attribuer de nouveaux paramètres et de nouvelles valeurs. Lorsque vous exécutez cette commande, toutes les valeurs de paramètre précédemment définies sont supprimées avant que les nouvelles valeurs ne soient attribuées. Seul un super-utilisateur peut modifier un fournisseur d'identité.

Syntaxe

```
ALTER IDENTITY PROVIDER identity_provider_name
[PARAMETERS parameter_string]
[NAMESPACE namespace]
[IAM_ROLE iam_role]
[AUTO_CREATE_ROLES
    [ TRUE [ { INCLUDE | EXCLUDE } GROUPS LIKE filter_pattern] |
      FALSE
    ]
[DISABLE | ENABLE]
```

Paramètres

identity_provider_name

Nom du nouveau fournisseur d'identité. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

parameter_string

Chaîne contenant un objet JSON correctement formaté qui contient des paramètres et des valeurs requis pour le fournisseur d'identité spécifique.

espace de nom

L'espace de noms de l'organisation.

iam_role

Rôle IAM qui fournit des autorisations pour la connexion à IAM Identity Center. Ce paramètre est applicable uniquement lorsque le type de fournisseur d'identité est. AWSIDC

auto_create_roles

Active ou désactive la fonctionnalité de création automatique des rôles. Si la valeur est VRAI, Amazon Redshift active la fonctionnalité de création automatique des rôles. Si la valeur est FALSE, Amazon Redshift désactive la fonctionnalité de création automatique des rôles. Si la valeur de ce paramètre n'est pas spécifiée, Amazon Redshift détermine la valeur selon la logique suivante :

- S'il `AUTO_CREATE_ROLES` est fourni mais que la valeur n'est pas spécifiée, la valeur est définie sur TRUE.
- Si `AUTO_CREATE_ROLES` ce n'est pas le cas et que le fournisseur d'identité l'est AWSIDC, la valeur est définie sur FALSE.
- Si `AUTO_CREATE_ROLES` ce n'est pas le cas et que le fournisseur d'identité est Azure, la valeur est définie sur TRUE.

Pour inclure des groupes, spécifiez`INCLUDE`. La valeur par défaut est vide, ce qui signifie inclure tous les groupes si cette option `AUTO_CREATE_ROLES` est activée.

Pour exclure des groupes, spécifiez`EXCLUDE`. La valeur par défaut est vide, ce qui signifie qu'il ne faut exclure aucun groupe si cette option `AUTO_CREATE_ROLES` est activée.

filter_pattern

Expression de caractères UTF-8 valide avec un modèle correspondant aux noms de groupes. L'option LIKE effectue une mise en correspondance sensible à la casse qui prend en charge les métacaractères de mise en correspondance de modèle suivants :

Métacaractère	Description
`%`	Met en correspondance une séquence de zéro ou plusieurs caractères.
`_`	Met en correspondance un seul caractère.

Si filter_pattern ne contient pas de métacaractères, le modèle représente uniquement la chaîne elle-même ; dans ce cas, LIKE a la même fonction que l'opérateur d'égalité.

filter_pattern prend en charge les caractères suivants :

- Caractères alphabétiques majuscules et minuscules (A-Z et a-z)
- Chiffres (0-9)
- Les caractères spéciaux suivants :

```
_ % ^ * + ? { } , $
```

DÉSACTIVER ou ACTIVER

Active ou désactive un fournisseur d'identité. La valeur par défaut est ENABLE.

Exemples

L'exemple suivant montre comment modifier un fournisseur d'identité nommé oauth_standard. Cela s'applique spécifiquement aux cas où Microsoft Azure AD est le fournisseur d'identité.

```
ALTER IDENTITY PROVIDER oauth_standard
PARAMETERS '{"issuer":"https://sts.windows.net/2sdfdsf-d475-420d-b5ac-667adad7c702/",
"client_id":"87f4aa26-78b7-410e-bf29-57b39929ef9a",
"client_secret":"BUAH~ewrqewrqwerUUY^%tHe1oNZShoiU7",
"audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift"]
}'
```

L'exemple suivant montre comment définir l'espace de noms du fournisseur d'identité. Cela peut s'appliquer à Microsoft Azure AD, s'il suit une instruction comme dans l'exemple précédent, ou à un autre fournisseur d'identité. Cela peut également s'appliquer au cas où vous connectez un cluster provisionné Amazon Redshift existant ou un groupe de travail Amazon Redshift Serverless à IAM Identity Center, si vous avez établi une connexion via une application gérée.

```
ALTER IDENTITY PROVIDER "my-redshift-idc-application"
NAMESPACE 'MYCO';
```

L'exemple suivant définit le rôle IAM et fonctionne dans le cas d'utilisation pour configurer l'intégration de Redshift avec IAM Identity Center.

```
ALTER IDENTITY PROVIDER "my-redshift-idc-application"
IAM_ROLE 'arn:aws:iam::123456789012:role/myadministratorrole';
```

Pour plus d'informations sur la configuration d'une connexion à IAM Identity Center depuis Redshift, voir Connect Redshift à IAM Identity Center pour offrir aux utilisateurs une expérience d'authentification unique.

Désactivation d'un fournisseur d'identité

L'exemple d'instruction suivant montre comment désactiver un fournisseur d'identité. Lorsqu'il est désactivé, les utilisateurs fédérés du fournisseur d'identité ne peuvent pas se connecter au cluster tant qu'il n'est pas réactivé.

```
ALTER IDENTITY PROVIDER "redshift-idc-app" DISABLE;
```

MODIFIER LA POLITIQUE DE MASQUAGE

Modifie une politique de masquage dynamique des données existante. Pour plus d'informations sur le masquage dynamique des données, consultez Masquage dynamique des données.

Les superutilisateurs et les utilisateurs ou rôles qui disposent du rôle sys:secadmin peuvent modifier une politique de masquage.

Syntaxe

```
ALTER MASKING POLICY policy_name
   USING (masking_expression);
```

Paramètres

policy_name

Nom de la politique de masquage. Il doit s'agir du nom d'une politique de masquage qui existe déjà dans la base de données.

masking_expression

Expression SQL utilisée pour transformer les colonnes cibles. Elle peut être écrite à l'aide de fonctions de manipulation des données, telles que des fonctions de manipulation de chaînes, ou en conjonction avec des fonctions définies par l'utilisateur écrites en SQL, Python ou avec AWS Lambda.

L'expression doit correspondre aux colonnes et aux types de données de l'expression originale. Par exemple, si les colonnes d'entrée de la politique de masquage originale sont `sample_1`

FLOAT et sample_2 VARCHAR(10), vous ne pourrez pas modifier la politique de masquage pour prendre une troisième colonne, ou faire en sorte que la politique prenne une instruction FLOAT et une instruction BOOLEAN. Si vous utilisez une constante comme expression de masquage, vous devez la convertir explicitement en un type correspondant au type d'entrée.

Vous devez disposer de l'autorisation USAGE sur toutes les fonctions définies par l'utilisateur que vous utilisez dans l'expression du masquage.

ALTER MATERIALIZED VIEW

Modifie les attributs d'une vue matérialisée.

Syntaxe

```
ALTER MATERIALIZED VIEW mv_name
{
AUTO REFRESH { YES | NO }
| ALTER DISTKEY column_name
| ALTER DISTSTYLE ALL
| ALTER DISTSTYLE EVEN
| ALTER DISTSTYLE KEY DISTKEY column_name
| ALTER DISTSTYLE AUTO
| ALTER [COMPOUND] SORTKEY ( column_name [,...] )
| ALTER SORTKEY AUTO
| ALTER SORTKEY NONE
| ROW LEVEL SECURITY { ON | OFF } [ CONJUNCTION TYPE { AND | OR } ] [FOR DATASHARES]
};
```

Paramètres

mv_name

Nom de la vue matérialisée à modifier.

AUTO REFRESH { YES | NO }

Clause qui active ou désactive l'actualisation automatique d'une vue matérialisée. Pour en savoir plus sur l'actualisation automatique des vues matérialisées, consultez Actualisation d'une vue matérialisée.

ALTER DISTSTYLE ALL

Clause qui modifie le style de distribution existant d'une relation avec`ALL`. Éléments à prendre en compte :

- Un ALTER DISTSTYLE, un ALTER SORTKEY et un VACUUM ne peuvent pas être exécutés simultanément sur la même relation.
  - Si VACUUM est en cours d'exécution, l'exécution de ALTER DISTSTYLE ALL renvoie une erreur.
  - Si ALTER DISTSTYLE ALL est en cours d'exécution, aucun aspirateur en arrière-plan ne démarre sur une relation.
- La commande ALTER DISTSTYLE ALL n'est pas prise en charge pour les relations avec des clés de tri entrelacées et des tables temporaires.
- Si le style de distribution était précédemment défini comme AUTO, la relation n'est plus éligible à l'optimisation automatique des tables.

Pour plus d'informations sur DISTSTYLE ALL, rendez-vous sur. CREATE MATERIALIZED VIEW

ALTER DISTSTYLE EVEN

Clause qui modifie le style de distribution existant d'une relation avec`EVEN`. Éléments à prendre en compte :

- Un ALTER DISTSYTLE, un ALTER SORTKEY et un VACUUM ne peuvent pas être exécutés simultanément sur la même relation.
  - Si VACUUM est en cours d'exécution, l'exécution d'ALTER DISTSTYLE EVEN renvoie une erreur.
  - Si ALTER DISTSTYLE EVEN est en cours d'exécution, un arrière-plan vide ne démarre pas sur une relation.
- La commande ALTER DISTSTYLE EVEN n'est pas prise en charge pour les relations avec des clés de tri entrelacées et des tables temporaires.
- Si le style de distribution était précédemment défini comme AUTO, la relation n'est plus éligible à l'optimisation automatique des tables.

Pour plus d'informations sur DISTSTYLE EVEN, rendez-vous sur. CREATE MATERIALIZED VIEW

ALTER DISTKEY nom_colonne ou ALTER DISTSTYLE KEY DISTKEY nom_colonne

Clause qui modifie la colonne utilisée comme clé de distribution d'une relation. Éléments à prendre en compte :

- VACUUM et ALTER DISTKEY ne peuvent pas être exécutés simultanément sur la même relation.
  - Si VACUUM est déjà en cours d'exécution, ALTER DISTKEY renvoie une erreur.
  - Si ALTER DISTKEY est en cours d'exécution, le vacuum en arrière-plan ne démarre pas sur une relation.
  - Si ALTER DISTKEY est en cours d'exécution, l'opération VACCUM au premier plan renvoie une erreur.
- Vous ne pouvez exécuter qu'une seule commande ALTER DISTKEY sur une relation à la fois.
- La commande ALTER DISTKEY n'est pas prise en charge pour les relations avec des clés de tri entrelacées.
- Si le style de distribution était précédemment défini comme AUTO, la relation n'est plus éligible à l'optimisation automatique des tables.

Lorsque DISTSTYLE KEY est spécifié, les données sont distribuées par les valeurs figurant dans la colonne DISTKEY. Pour plus d'informations sur DISTSTYLE, rendez-vous sur. CREATE MATERIALIZED VIEW

ALTER DISTSTYLE AUTO

Clause qui modifie le style de distribution existant d'une relation en AUTO.

Lorsque vous modifiez un style de distribution en AUTO, le style de distribution de la relation est défini comme suit :

- Une petite relation avec DISTSTYLE ALL est convertie en AUTO (ALL).
- Une petite relation avec DISTSTYLE EVEN est convertie en AUTO (ALL).
- Une petite relation avec DISTSTYLE KEY est convertie en AUTO (ALL).
- Une relation importante avec DISTSTYLE ALL est convertie en AUTO (EVEN).
- Une relation étendue avec DISTSTYLE EVEN est convertie en AUTO (EVEN).
- Une relation importante avec DISTSTYLE KEY est convertie en AUTO (KEY) et le DISTKEY est préservé. Dans ce cas, Amazon Redshift n'apporte aucune modification à la relation.

Si Amazon Redshift détermine qu'un nouveau style ou une nouvelle clé de distribution améliorera les performances des requêtes, Amazon Redshift peut modifier le style ou la clé de distribution de

votre relation à l'avenir. Par exemple, Amazon Redshift peut convertir une relation dont la valeur DISTSTYLE est AUTO (KEY) en AUTO (EVEN), ou vice versa. Pour plus d'informations sur le comportement en cas de modification des clés de distribution, notamment sur la redistribution des données et les verrouillages, consultez les recommandations d'Amazon Redshift Advisor.

Pour plus d'informations sur DISTSTYLE AUTO, rendez-vous sur. CREATE MATERIALIZED VIEW

Pour afficher le style de distribution d'une relation, interrogez la vue du catalogue du système SVV_TABLE_INFO. Pour plus d'informations, consultez la section concernant SVV_TABLE_INFO. Pour consulter les recommandations d'Amazon Redshift Advisor en matière de relations, interrogez la vue du catalogue du système SVV_ALTER_TABLE_RECOMMENDATIONS. Pour plus d'informations, consultez la section concernant SVV_ALTER_TABLE_RECOMMENDATIONS. Pour afficher les actions effectuées par Amazon Redshift, interrogez la vue catalogue système SVL_AUTO_WORKER_ACTION. Pour plus d'informations, consultez la section concernant SVL_AUTO_WORKER_ACTION.

ALTER [COMPOUND] SORTKEY (column_name [,...])

Clause qui modifie ou ajoute la clé de tri utilisée pour une relation. ALTER SORTKEY n'est pas pris en charge pour les tables temporaires.

Lorsque vous modifiez une clé de tri, l'encodage par compression des colonnes de la nouvelle clé de tri ou de la clé de tri originale peut changer. Si aucun codage n'est explicitement défini pour la relation, Amazon Redshift attribue automatiquement des codages de compression comme suit :

- Les colonnes qui sont définies comme des clés de tri se voient attribuer une compression RAW.
- Les colonnes qui sont définies comme des types de données BOOLEAN, REAL ou DOUBLE PRECISION se voient attribuer une compression RAW.
- Les colonnes définies comme SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ sont soumises à une compression. AZ64
- Les colonnes définies comme CHAR ou VARCHAR sont affectées à la compression LZO.

Éléments à prendre en compte :

- Vous pouvez définir un maximum de 400 colonnes pour une clé de tri par relation.
- Vous pouvez modifier une clé de tri entrelacée en clé de tri composée ou en aucune clé de tri. Toutefois, vous ne pouvez pas modifier une clé de tri composée par une clé de tri entrelacée.
- Si la clé de tri était précédemment définie comme AUTO, la relation n'est plus candidate à l'optimisation automatique des tables.

- Amazon Redshift recommande d'utiliser l'encodage RAW (sans compression) pour les colonnes définies comme des clés de tri. Lorsque vous modifiez une colonne pour la choisir comme clé de tri, la compression de la colonne passe en compression RAW (sans compression). Cela peut augmenter la quantité de stockage requise par la relation. L'augmentation de la taille de la relation dépend de la définition spécifique de la relation et du contenu de la relation. Pour plus d'informations sur la compression, consultezencodages de compression.

Lorsque des données sont chargées dans une relation, elles sont chargées dans l'ordre de la clé de tri. Lorsque vous modifiez la clé de tri, Amazon Redshift modifie l'ordre des données. Pour plus d'informations sur SORTKEY, rendez-vous sur. CREATE MATERIALIZED VIEW

ALTER TRUTKEY AUTO

Clause qui modifie ou ajoute la clé de tri de la relation cible à AUTO. ALTER SORTKEY AUTO n'est pas pris en charge pour les tables temporaires.

Lorsque vous remplacez une clé de tri par AUTO, Amazon Redshift préserve la clé de tri existante de la relation.

Si Amazon Redshift détermine qu'une nouvelle clé de tri améliorera les performances des requêtes, Amazon Redshift peut modifier la clé de tri de votre relation à l'avenir.

Pour plus d'informations sur SORTKEY AUTO, rendez-vous sur. CREATE MATERIALIZED VIEW

Pour afficher la clé de tri d'une relation, interrogez la vue du catalogue du système SVV_TABLE_INFO. Pour plus d'informations, consultez la section concernant SVV_TABLE_INFO. Pour consulter les recommandations d'Amazon Redshift Advisor en matière de relations, interrogez la vue du catalogue du système SVV_ALTER_TABLE_RECOMMENDATIONS. Pour plus d'informations, consultez la section concernant SVV_ALTER_TABLE_RECOMMENDATIONS. Pour afficher les actions effectuées par Amazon Redshift, interrogez la vue catalogue système SVL_AUTO_WORKER_ACTION. Pour plus d'informations, consultez la section concernant SVL_AUTO_WORKER_ACTION.

ALTER SORTKEY NONE

Clause qui supprime la clé de tri de la relation cible.

Si la clé de tri était précédemment définie comme AUTO, la relation n'est plus candidate à l'optimisation automatique des tables.

ROW LEVEL SECURITY { ON | OFF } [CONJUNCTION TYPE { AND | OR }] [FOR DATASHARES]

Clause qui active ou désactive la sécurité au niveau des lignes pour une relation.

Lorsque la sécurité au niveau des lignes est activée pour une relation, vous pouvez lire uniquement les lignes auxquelles cette politique vous autorise à accéder. Si aucune politique ne vous accorde l'accès à la relation, vous ne pouvez voir aucune ligne pour la relation. Seuls les super-utilisateurs et les utilisateurs ou les rôles dotés du rôle `sys:secadmin` peuvent définir la clause ROW LEVEL SECURITY. Pour de plus amples informations, veuillez consulter Sécurité au niveau des lignes.

- [CONJUNCTION TYPE { AND | OR }]

  Clause qui vous permet de choisir le type de conjonction de la politique de sécurité au niveau des lignes pour une relation. Lorsque plusieurs politiques de sécurité au niveau des lignes sont associées à une relation, vous pouvez les combiner avec la clause AND ou OR. Par défaut, Amazon Redshift associe les politiques RLS à la clause AND. Les super-utilisateurs, utilisateurs ou rôles disposant du rôle `sys:secadmin` peuvent utiliser cette clause pour définir le type de conjonction de la politique de sécurité au niveau des lignes pour une relation. Pour de plus amples informations, veuillez consulter Association de plusieurs politiques par utilisateur.

- FOR DATASHARES

  Clause qui détermine s'il est possible d'accéder à une relation protégée par RLS sur les unités de partage des données. Par défaut, une relation protégée par RLS n'est pas accessible sur une unité de partage des données. Une commande ALTER MATERIALIZED VIEW ROW LEVEL SECURITY exécutée avec cette clause affecte uniquement la propriété d'accessibilité de l'unité de partage des données de la relation. La propriété ROW LEVEL SECURITY n'est pas modifiée.

  Si vous rendez accessible une relation protégée par RLS via des unités de partage des données, la relation n'est pas sécurisée au niveau des lignes dans la base de données de l'unité de partage des données côté consommateur. La relation conserve sa propriété RLS du côté producteur.

Exemples

L'exemple suivant active la vue matérialisée `tickets_mv` à actualiser automatiquement.

```
ALTER MATERIALIZED VIEW tickets_mv AUTO REFRESH YES
```

Exemples DISTSTYLE et SORTKEY pour ALTER MATERIALIZED VIEW

Les exemples présentés dans cette rubrique vous montrent comment effectuer les modifications DISTSTYLE et SORTKEY à l'aide de ALTER MATERIALIZED VIEW.

Les exemples de requêtes suivants montrent comment modifier une colonne DISTSTYLE KEY DISTKEY à l'aide d'un exemple de table de base :

```
CREATE TABLE base_inventory(
  inv_date_sk int4 NOT NULL,
  inv_item_sk int4 NOT NULL,
  inv_warehouse_sk int4 NOT NULL,
  inv_quantity_on_hand int4
);

INSERT INTO base_inventory VALUES(1,1,1,1);

CREATE materialized VIEW inventory diststyle even AS SELECT * FROM base_inventory;
SELECT "table", diststyle FROM svv_table_info WHERE "table" = 'inventory';

ALTER materialized VIEW inventory ALTER diststyle KEY distkey inv_warehouse_sk;
SELECT "table", diststyle FROM svv_table_info WHERE "table" = 'inventory';

ALTER materialized VIEW inventory ALTER distkey inv_item_sk;
SELECT "table", diststyle FROM svv_table_info WHERE "table" = 'inventory';

DROP TABLE base_inventory CASCADE;
```

Modifiez une vue matérialisée en DISTSTYLE ALL :

```
CREATE TABLE base_inventory(
  inv_date_sk int4 NOT NULL,
  inv_item_sk int4 NOT NULL,
  inv_warehouse_sk int4 NOT NULL,
  inv_quantity_on_hand int4
);

INSERT INTO base_inventory VALUES(1,1,1,1);

CREATE materialized VIEW inventory diststyle even AS SELECT * FROM base_inventory;
SELECT "table", diststyle FROM svv_table_info WHERE "table" = 'inventory';
```

```
ALTER MATERIALIZED VIEW inventory ALTER diststyle ALL;
SELECT "table", diststyle FROM svv_table_info WHERE "table" = 'inventory';

DROP TABLE base_inventory CASCADE;
```

Les commandes suivantes présentent des exemples ALTER MATERIALIZED VIEW SORTKEY utilisant un exemple de table de base :

```
CREATE TABLE base_inventory (c0 int, c1 int);

INSERT INTO base_inventory VALUES(1,1);

CREATE materialized VIEW inventory interleaved sortkey(c0, c1) AS SELECT * FROM
 base_inventory;
SELECT "table", sortkey1 FROM svv_table_info WHERE "table" = 'inventory';

ALTER materialized VIEW inventory ALTER sortkey(c0, c1);
SELECT "table", diststyle, sortkey_num FROM svv_table_info WHERE "table" = 'inventory';

ALTER materialized VIEW inventory ALTER sortkey NONE;
SELECT "table", diststyle, sortkey_num FROM svv_table_info WHERE "table" = 'inventory';

ALTER materialized VIEW inventory ALTER sortkey(c0);
SELECT "table", diststyle, sortkey_num FROM svv_table_info WHERE "table" = 'inventory';

DROP TABLE base_inventory CASCADE;
```

ALTER RLS POLICY

Modifie une politique de sécurité existante au niveau des lignes sur une table.

Les super-utilisateurs et les utilisateurs ou les rôles dotés du rôle `sys:secadmin` peuvent modifier une politique.

Syntaxe

```
ALTER RLS POLICY policy_name
USING ( using_predicate_exp );
```

Paramètres

policy_name

Nom de la politique .

USING (using_predicate_exp)

Spécifie un filtre qui est appliqué à la clause WHERE de la requête. Amazon Redshift applique un prédicat de politique avant les prédicats utilisateur au niveau de la requête. Par exemple, **`current_user = 'joe' and price > 10`** permet à Joe de ne voir que les enregistrements dont le prix est supérieur à 10 $.

L'expression a accès aux variables déclarées dans la clause WITH de l'instruction CREATE RLS POLICY qui a été utilisée pour créer la politique nommée policy_name.

Exemples

L'exemple suivant modifie une politique RLS.

```
-- First create an RLS policy that limits access to rows where catgroup is 'concerts'.
CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'concerts');

-- Then, alter the RLS policy to only show rows where catgroup is 'piano concerts'.
ALTER RLS POLICY policy_concerts
USING (catgroup = 'piano concerts');
```

ALTER ROLE

Renomme un rôle ou modifie le propriétaire. Pour obtenir une liste des rôles définis par le système Amazon Redshift, consultez the section called "Rôles définis par le système Amazon Redshift".

Autorisations requises

Voici les autorisations requises pour ALTER ROLE :

- Superuser
- Utilisateurs disposant des autorisations ALTER ROLE

Syntaxe

```
ALTER ROLE role [ WITH ]
  { { RENAME TO role } | { OWNER TO user_name } }[, ...]
  [ EXTERNALID TO external_id ]
```

Paramètres

rôle

Nom du rôle à modifier.

RENAME TO

Nouveau nom du rôle.

OWNER TO user_name

Nouveau propriétaire du rôle.

EXTERNALID TO external_id

Nouvel ID externe pour le rôle, associé à un fournisseur d'identité. Pour plus d'informations, consultez Fédération de fournisseur d'identité natif pour Amazon Redshift.

Exemples

L'exemple suivant remplace le nom `sample_role1` d'un rôle par `sample_role2`.

```
ALTER ROLE sample_role1 RENAME TO sample_role2;
```

L'exemple suivant remplace le propriétaire du rôle.

```
ALTER ROLE sample_role1 WITH OWNER TO user1
```

La syntaxe de ALTER ROLE est semblable à celle de ALTER PROCEDURE, reprise ci-après.

```
ALTER PROCEDURE first_quarter_revenue(bigint, numeric) RENAME TO quarterly_revenue;
```

L'exemple suivant remplace le propriétaire d'une procédure par `etl_user`.

```
ALTER PROCEDURE quarterly_revenue(bigint, numeric) OWNER TO etl_user;
```

L'exemple suivant montre comment mettre à jour un rôle `sample_role1` avec un nouvel ID externe associé à un fournisseur d'identité.

```
ALTER ROLE sample_role1 EXTERNALID TO "XYZ456";
```

ALTER PROCEDURE

Renomme une procédure ou modifie le propriétaire. Le nom de la procédure et les types de données, ou signature, sont requis. Seul le propriétaire ou un super-utilisateur peut renommer une procédure. Seul un super-utilisateur peut changer le propriétaire d'une procédure.

Syntaxe

```
ALTER PROCEDURE sp_name [ ( [ [ argname ] [ argmode ] argtype [, ...] ] ) ]
    RENAME TO new_name
```

```
ALTER PROCEDURE sp_name [ ( [ [ argname ] [ argmode ] argtype [, ...] ] ) ]
    OWNER TO { new_owner | CURRENT_USER | SESSION_USER }
```

Paramètres

sp_name

Nom de la procédure à modifier. Spécifiez simplement le nom de la procédure dans le chemin de recherche actuel ou choisissez le format `schema_name.sp_procedure_name` pour utiliser un schéma spécifique.

[argname] [argmode] argtype

Liste de noms d'arguments, de modes d'argument et de types de données. Seuls les types de données d'entrée sont requis. Ils sont utilisés pour identifier la procédure stockée. Vous pouvez également indiquer la procédure complète utilisée pour créer la procédure, en incluant les paramètres d'entrée et de sortie avec leurs modes.

nouveau_nom

Nouveau nom pour la procédure stockée.

new_owner | CURRENT_USER | SESSION_USER

Nouveau propriétaire pour la procédure stockée.

Exemples

L'exemple suivant remplace le nom `first_quarter_revenue` d'une procédure par `quarterly_revenue`.

```
ALTER PROCEDURE first_quarter_revenue(volume INOUT bigint, at_price IN numeric,
 result OUT int) RENAME TO quarterly_revenue;
```

Cet exemple équivaut à l'exemple suivant :

```
ALTER PROCEDURE first_quarter_revenue(bigint, numeric) RENAME TO quarterly_revenue;
```

L'exemple suivant remplace le propriétaire d'une procédure par `etl_user`.

```
ALTER PROCEDURE quarterly_revenue(bigint, numeric) OWNER TO etl_user;
```

ALTER SCHEMA

Modifie la définition d'un schéma existant. Utilisez cette commande pour renommer un schéma ou en modifier le propriétaire. Par exemple, renommez un schéma existant pour en conserver une copie de sauvegarde lorsque vous prévoyez d'en créer une nouvelle version. Pour plus d'informations sur les schémas, consultez CREATE SCHEMA.

Pour afficher les quotas de schéma configurés, consultez SVV_SCHEMA_QUOTA_STATE.

Pour afficher les enregistrements où les quotas de schéma ont été dépassés, consultez STL_SCHEMA_QUOTA_VIOLATIONS.

Privilèges requis

Les privilèges suivants sont requis pour ALTER SCHEMA :

- Superuser

- Utilisateurs disposant du privilège ALTER SCHEMA
- Propriétaire du schéma

Lorsque vous modifiez le nom d'un schéma, notez que les objets utilisant l'ancien nom, tels que les procédures stockées ou les vues matérialisées, doivent être mis à jour pour utiliser le nouveau nom.

Syntaxe

```
ALTER SCHEMA schema_name
{
RENAME TO new_name |
OWNER TO new_owner |
QUOTA { quota [MB | GB | TB] | UNLIMITED }
}
```

Paramètres

nom_schéma

Nom du schéma de base de données à modifier.

RENAME TO

Clause qui renomme le schéma.

nouveau_nom

Nouveau nom du schéma. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

OWNER TO

Clause qui modifie le propriétaire du schéma.

nouveau_propriétaire

Nouveau propriétaire du schéma.

QUOTA

Quantité maximale d'espace disque que le schéma spécifié peut utiliser. Cet espace correspond à la taille collective de toutes les tables sous le schéma spécifié. Amazon Redshift convertit la

valeur sélectionnée en mégaoctets. Le gigaoctet est l'unité de mesure par défaut lorsque vous ne spécifiez pas de valeur.

Pour plus d'informations sur la configuration des quotas de schéma, consultez CREATE SCHEMA.

Exemples

L'exemple suivant renomme le schéma SALES en US_SALES.

```
alter schema sales
rename to us_sales;
```

L'exemple suivant attribue la propriété du schéma US_SALES à l'utilisateur DWUSER.

```
alter schema us_sales
owner to dwuser;
```

L'exemple suivant modifie le quota à 300 Mo et le supprime.

```
alter schema us_sales QUOTA 300 GB;
alter schema us_sales QUOTA UNLIMITED;
```

ALTER SYSTEM

Modifie une option de configuration au niveau du système pour le cluster Amazon Redshift ou le groupe de travail Redshift sans serveur.

Privilèges requis

La commande ALTER SYSTEM peut être exécutée par l'un des types d'utilisateurs suivants :

- Superuser
- Utilisateur administrateur

Syntaxe

```
ALTER SYSTEM SET system-level-configuration = {true| t | on | false | f | off}
```

Paramètres

system-level-configuration

Configuration au niveau du système. Valeurs valides : `data_catalog_auto_mount` et `metadata_security`.

{true| t | on | false | f | off}

Valeur permettant d'activer ou de désactiver la configuration au niveau du système. Les valeurs `true`, `t` et `on` indiquent que la configuration doit être activée. Les valeurs `false`, `f` et `off` indiquent que la configuration doit être désactivée.

Notes d'utilisation

Pour un cluster provisionné, toute modification apportée à `data_catalog_auto_mount` prend effet au prochain redémarrage du cluster. Pour en savoir plus, consultez Redémarrage d'un cluster dans Amazon Redshift Management Guide.

Pour un groupe de travail sans serveur, les modifications apportées à `data_catalog_auto_mount` ne prennent pas effet immédiatement.

Exemples

L'exemple suivant active le montage automatique du AWS Glue Data Catalog.

```
ALTER SYSTEM SET data_catalog_auto_mount = true;
```

L'exemple suivant active la sécurité des métadonnées.

```
ALTER SYSTEM SET metadata_security = true;
```

Définition d'un espace de noms d'identité par défaut

Cet exemple est spécifique à l'utilisation d'un fournisseur d'identité. Vous pouvez intégrer Redshift à IAM Identity Center et à un fournisseur d'identité afin de centraliser la gestion des identités pour Redshift et d'autres services. AWS

L'exemple suivant montre comment définir l'espace de noms d'identité par défaut pour le système. Cela simplifie ensuite l'exécution des instructions GRANT et CREATE, car il n'est pas nécessaire d'inclure l'espace de noms comme préfixe pour chaque identité.

```
ALTER SYSTEM SET default_identity_namespace = 'MYCO';
```

Après avoir exécuté la commande, vous pouvez exécuter des instructions telles que les suivantes :

```
GRANT SELECT ON TABLE mytable TO alice;

GRANT UPDATE ON TABLE mytable TO salesrole;

CREATE USER bob password 'md50c983d1a624280812631c5389e60d48c';
```

La définition de l'espace de noms d'identité par défaut a pour effet que chaque identité ne l'exige pas comme préfixe. Dans cet exemple, `alice` est remplacé par`MYCO:alice`. Cela se produit avec n'importe quelle identité incluse. Pour plus d'informations sur l'utilisation d'un fournisseur d'identité avec Redshift, consultez Connect Redshift à IAM Identity Center pour offrir aux utilisateurs une expérience d'authentification unique.

Pour plus d'informations sur les paramètres relatifs à la configuration de Redshift avec IAM Identity Center, consultez et. SET ALTER IDENTITY PROVIDER

ALTER TABLE

Cette commande modifie la définition d'une table Amazon Redshift ou d'une table externe Amazon Redshift Spectrum. Cette commande met à jour les valeurs et les propriétés définies par CREATE TABLE ou CREATE EXTERNAL TABLE. Vous pouvez utiliser ALTER TABLE sur une vue pour la sécurité au niveau des lignes (RLS).

Vous ne pouvez pas exécuter ALTER TABLE sur une table externe au sein d'un bloc de transaction (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

ALTER TABLE verrouille la table pour les opérations de lecture et d'écriture jusqu'à la fin de la transaction englobant l'opération ALTER TABLE, sauf s'il est indiqué spécifiquement dans la documentation que vous pouvez interroger des données ou effectuer d'autres opérations sur la table pendant sa modification.

Privilèges requis

L'utilisateur qui modifie une table doit disposer des privilèges appropriés pour que la commande aboutisse. Selon la commande ALTER TABLE, l'un des privilèges suivants est requis.

- Superuser
- Utilisateurs disposant du privilège ALTER TABLE
- Propriétaire de table disposant du privilège USAGE sur le schéma

Syntaxe

```
ALTER TABLE table_name
{
ADD table_constraint
| DROP CONSTRAINT constraint_name [ RESTRICT | CASCADE ]
| OWNER TO new_owner
| RENAME TO new_name
| RENAME COLUMN column_name TO new_name
| ALTER COLUMN column_name TYPE updated_varchar_data_type_size
| ALTER COLUMN column_name ENCODE new_encode_type
| ALTER COLUMN column_name ENCODE encode_type,
| ALTER COLUMN column_name ENCODE encode_type, .....;
| ALTER DISTKEY column_name
| ALTER DISTSTYLE ALL
| ALTER DISTSTYLE EVEN
| ALTER DISTSTYLE KEY DISTKEY column_name
| ALTER DISTSTYLE AUTO
| ALTER [COMPOUND] SORTKEY ( column_name [,...] )
| ALTER SORTKEY AUTO
| ALTER SORTKEY NONE
| ALTER ENCODE AUTO
| ADD [ COLUMN ] column_name column_type
  [ DEFAULT default_expr ]
  [ ENCODE encoding ]
  [ NOT NULL | NULL ]
  [ COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE } ] |
| DROP [ COLUMN ] column_name [ RESTRICT | CASCADE ]
| ROW LEVEL SECURITY { ON | OFF } [ CONJUNCTION TYPE { AND | OR } ] [ FOR DATASHARES ]
| MASKING { ON | OFF } FOR DATASHARES }

where table_constraint is:

[ CONSTRAINT constraint_name ]
{ UNIQUE ( column_name [, ... ] )
| PRIMARY KEY ( column_name [, ... ] )
| FOREIGN KEY (column_name [, ... ] )
   REFERENCES  reftable [ ( refcolumn ) ]}
```

```
The following options apply only to external tables:

SET LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file' }
| SET FILE FORMAT format |
| SET TABLE PROPERTIES ('property_name'='property_value')
| PARTITION ( partition_column=partition_value [, ...] )
  SET LOCATION { 's3://bucket/folder' |'s3://bucket/manifest_file' }
| ADD [IF NOT EXISTS]
    PARTITION ( partition_column=partition_value [, ...] ) LOCATION
 { 's3://bucket/folder' |'s3://bucket/manifest_file' }
    [, ... ]
| DROP PARTITION ( partition_column=partition_value [, ...] )
```

Pour réduire le temps d'exécution de la commande ALTER TABLE, vous pouvez combiner certaines clauses de la commande ALTER TABLE.

Amazon Redshift prend en charge les combinaisons suivantes des clauses ALTER TABLE :

```
ALTER TABLE tablename ALTER SORTKEY (column_list), ALTER DISTKEY column_Id;
ALTER TABLE tablename ALTER DISTKEY column_Id, ALTER SORTKEY (column_list);
ALTER TABLE tablename ALTER SORTKEY (column_list), ALTER DISTSTYLE ALL;
ALTER TABLE tablename ALTER DISTSTYLE ALL, ALTER SORTKEY (column_list);
```

Paramètres

table_name

Nom de la table à modifier. Spécifiez simplement le nom de la table ou choisissez le format nom_schéma.nom_table pour utiliser un schéma spécifique. Les tables externes doivent être qualifiées à l'aide d'un nom de schéma externe. Vous pouvez aussi spécifier un nom de vue si vous utilisez l'instruction ALTER TABLE pour renommer une vue ou modifier son propriétaire. La longueur maximale d'un nom de table est de 127 octets ; les noms plus longs sont tronqués à 127 octets. Vous pouvez utiliser des caractères multioctets UTF-8 jusqu'à un maximum de quatre octets. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

ADD contrainte_table

Clause qui ajoute la contrainte spécifiée à la table. Pour obtenir les descriptions des valeurs de contrainte_table valides, consultez CREATE TABLE.

> **Note**
>
> Vous ne pouvez pas ajouter une contrainte de clé primaire à une colonne acceptant la valeur null. Si la colonne a été créée à l'origine avec la contrainte NOT NULL, vous pouvez ajouter la contrainte de clé primaire.

DROP CONSTRAINT nom_contrainte

Clause qui supprime la contrainte nommée de la table. Pour supprimer une contrainte, spécifiez son nom et non son type. Pour afficher le nom des contraintes de la table, exécutez la requête suivante.

```
select constraint_name, constraint_type
from information_schema.table_constraints;
```

RESTRICT

Clause qui supprime uniquement la contrainte spécifiée. RESTRICT est une option de DROP CONSTRAINT. RESTRICT ne peut pas être utilisée avec CASCADE.

CASCADE

Clause qui supprime la contrainte spécifiée ainsi que tout ce qui en dépend. CASCADE est une option de DROP CONSTRAINT. CASCADE ne peut pas être utilisée avec RESTRICT.

OWNER TO nouveau_propriétaire

Clause qui modifie le propriétaire de la table (ou de la vue) avec la valeur nouveau_propriétaire.

RENAME TO nouveau_nom

Clause qui renomme une table (ou une vue) selon la valeur spécifiée dans nouveau_nom. La longueur maximale d'un nom de table est de 127 octets ; les noms plus longs sont tronqués à 127 octets.

Vous ne pouvez pas renommer une table permanente en un nom commençant par « # ». Un nom de table commençant par « # » indique une table temporaire.

Vous ne pouvez pas renommer une table externe.

ALTER COLUMN column_name TYPE updated_varchar_data_type_size

Clause qui modifie la taille d'une colonne définie en tant que type de données VARCHAR. Cette clause permet uniquement de modifier la taille d'un type de données VARCHAR. Prenez en compte les restrictions suivantes :

- Vous ne pouvez pas modifier une colonne avec les codages de compression BYTEDICT, RUNLENGTH ou K. TEXT255 TEXT32
- Vous ne pouvez pas réduire la taille en dessous de la taille maximale des données existantes.
- Vous ne pouvez pas modifier des colonnes avec des valeurs par défaut.
- Vous ne pouvez pas modifier des colonnes avec UNIQUE, PRIMARY KEY ou FOREIGN KEY.
- Vous ne pouvez pas modifier des colonnes dans un bloc de transaction (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

ALTER COLUMN column_name ENCODE new_encode_type

Clause qui modifie l'encodage de compression d'une colonne. Si vous spécifiez le codage de compression pour une colonne, la table n'est plus définie sur ENCODE AUTO. Pour obtenir des informations sur l'encodage de la compression, consultez Compression de colonnes pour réduire la taille des données stockées.

Lorsque vous modifiez l'encodage de compression pour une colonne, la table reste disponible à des fins d'interrogation.

Prenez en compte les restrictions suivantes :

- Vous ne pouvez pas modifier une colonne avec le même encodage que celui actuellement défini pour la colonne.
- Vous ne pouvez pas modifier l'encodage d'une colonne dans une table avec une clé de tri entrelacée.

ALTER COLUMN column_name ENCODE encode_type, ALTER COLUMN column_name ENCODE encode_type, ...;

Clause qui modifie le codage de compression de plusieurs colonnes dans une seule commande. Pour obtenir des informations sur l'encodage de la compression, consultez Compression de colonnes pour réduire la taille des données stockées.

Lorsque vous modifiez l'encodage de compression pour une colonne, la table reste disponible à des fins d'interrogation.

Prenez en compte les restrictions suivantes :

- Vous ne pouvez pas modifier une colonne pour un type de codage identique ou différent plusieurs fois dans une seule commande.
- Vous ne pouvez pas modifier une colonne avec le même encodage que celui actuellement défini pour la colonne.
- Vous ne pouvez pas modifier l'encodage d'une colonne dans une table avec une clé de tri entrelacée.

ALTER DISTSTYLE ALL

Clause qui modifie le style de distribution existant d'une table en `ALL`. Éléments à prendre en compte :

- ALTER DISTSYTLE, ALTER SORTKEY et VACUUM ne peuvent pas s'exécuter simultanément sur la même table.
  - Si VACUUM est en cours d'exécution, l'exécution de ALTER DISTSTYLE ALL renvoie une erreur.
  - Si ALTER DISTSTYLE ALL est en cours d'exécution, une opération VACCUM en arrière-plan ne démarre pas sur une table.
- La commande ALTER DISTSTYLE ALL n'est pas prise en charge pour les tables ayant des clés de tri entrelacées et les tables temporaires.
- Si le style de distribution était précédemment défini sur AUTO, la table n'est plus candidate à l'optimisation automatique.

Pour plus d'informations sur DISTSTYLE ALL, consultez CREATE TABLE.

ALTER DISTSTYLE EVEN

Clause qui modifie le style de distribution existant d'une table en `EVEN`. Éléments à prendre en compte :

- ALTER DISTSYTLE, ALTER SORTKEY et VACUUM ne peuvent pas être exécutés simultanément sur la même table.
  - Si VACUUM est en cours d'exécution, l'exécution d'ALTER DISTSTYLE EVEN renvoie une erreur.
  - Si ALTER DISTSTYLE EVEN est en cours d'exécution, une opération VACUUM en arrière-plan ne démarre pas sur une table.
- La commande ALTER DISTSTYLE EVEN n'est pas prise en charge pour les tables ayant des clés de tri entrelacées et des tables temporaires.

- Si le style de distribution était précédemment défini sur AUTO, la table n'est plus candidate à l'optimisation automatique.

Pour plus d'informations sur DISTSTYLE ALL, consultez CREATE TABLE.

ALTER DISTKEY nom_colonne ou ALTER DISTSTYLE KEY DISTKEY nom_colonne

Clause qui modifie la colonne utilisée en tant que clé de distribution d'une table. Éléments à prendre en compte :

- VACUUM et ALTER DISTKEY ne peuvent pas s'exécuter simultanément sur la même table.
  - Si VACUUM est déjà en cours d'exécution, ALTER DISTKEY renvoie une erreur.
  - Si ALTER DISTKEY est en cours d'exécution, l'opération VACCUM en arrière-plan ne démarre pas sur une table.
  - Si ALTER DISTKEY est en cours d'exécution, l'opération VACCUM au premier plan renvoie une erreur.
- Vous pouvez exécuter une seule commande ALTER DISTKEY sur une table simultanément.
- La commande ALTER DISTKEY n'est pas prise en charge pour les tables ayant des clés de tri entrelacées.
- Si le style de distribution était précédemment défini sur AUTO, la table n'est plus candidate à l'optimisation automatique.

Lorsque DISTSTYLE KEY est spécifié, les données sont distribuées par les valeurs figurant dans la colonne DISTKEY. Pour plus d'informations sur DISTSTYLE, consultez CREATE TABLE.

ALTER DISTSTYLE AUTO

Clause qui modifie le style de distribution existant d'une table en AUTO.

Lorsque vous modifiez un style de distribution sur AUTO, le style de distribution de la table est défini comme suit :

- Une petite table avec DISTSTYLE ALL est convertie en AUTO(ALL).
- Une petite table avec DISTSTYLE EVEN est convertie en AUTO(ALL).
- Une petite table avec DISTSTYLE KEY est convertie en AUTO(ALL).
- Une grande table avec DISTSTYLE ALL est convertie en AUTO(EVEN).
- Une grande table avec DISTSTYLE EVEN est convertie en AUTO(EVEN).
- Une grande table avec DISTSTYLE KEY est convertie en AUTO(KEY) et la DISTKEY est conservée. Dans ce cas, Amazon Redshift ne modifie pas la table.

Si Amazon Redshift détermine qu'un nouveau style de distribution ou une nouvelle clé améliorera les performances des requêtes, Amazon Redshift peut modifier le style de distribution ou la clé de votre table à l'avenir. Par exemple, Amazon Redshift peut convertir une table avec DISTSTYLE pour AUTO(KEY) en AUTO(EVEN), ou vice versa. Pour en savoir plus sur le comportement dans le cas où des clés de distribution sont modifiées, notamment sur le plan de la redistribution des données et des verrous, consultez Recommandations Amazon Redshift Advisor.

Pour plus d'informations sur DISTSTYLE AUTO, consultez CREATE TABLE.

Pour afficher le style de distribution appliqué à une table, interrogez la vue du catalogue système SVV_TABLE_INFO. Pour plus d'informations, consultez SVV_TABLE_INFO. Pour afficher les recommandations Amazon Redshift Advisor pour les tables, recherchez la vue catalogue système SVV_ALTER_TABLE_RECOMMENDATIONS. Pour plus d'informations, consultez SVV_ALTER_TABLE_RECOMMENDATIONS. Pour afficher les actions effectuées par Amazon Redshift, interrogez la vue catalogue système SVL_AUTO_WORKER_ACTION. Pour plus d'informations, consultez SVL_AUTO_WORKER_ACTION.

ALTER [COMPOUND] SORTKEY (column_name [,...])

Clause qui modifie ou ajoute la clé de tri utilisée pour une table. ALTER SORTKEY n'est pas pris en charge pour les tables temporaires.

Lorsque vous modifiez une clé de tri, l'encodage par compression des colonnes de la nouvelle clé de tri ou de la clé de tri originale peut changer. Si aucun encodage n'est explicitement défini pour la table, Amazon Redshift attribue automatiquement les codages de compression comme suit :

- Les colonnes qui sont définies comme des clés de tri se voient attribuer une compression RAW.
- Les colonnes qui sont définies comme des types de données BOOLEAN, REAL ou DOUBLE PRECISION se voient attribuer une compression RAW.
- Les colonnes définies comme SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ sont soumises à une compression. AZ64
- Les colonnes définies comme CHAR ou VARCHAR sont affectées à la compression LZO.

Éléments à prendre en compte :

- Vous pouvez définir 400 colonnes au maximum par table pour une clé de tri.
- Vous pouvez modifier une clé de tri entrelacée en clé de tri composée ou en aucune clé de tri. Toutefois, vous ne pouvez pas modifier une clé de tri composée par une clé de tri entrelacée.
- Si la clé de tri était précédemment définie sur AUTO, la table n'est plus candidate à l'optimisation automatique.

- Amazon Redshift recommande d'utiliser l'encodage RAW (sans compression) pour les colonnes définies comme des clés de tri. Lorsque vous modifiez une colonne pour la choisir comme clé de tri, la compression de la colonne passe en compression RAW (sans compression). Cela peut augmenter la quantité de stockage requise par la table. L'augmentation de la taille de la table dépend de la définition et du contenu spécifiques de la table. Pour plus d'informations sur la compression, consultez encodages de compression.

Les données sont chargées dans une table en fonction de la clé de tri. Lorsque vous modifiez la clé de tri, Amazon Redshift modifie l'ordre des données. Pour plus d'informations sur SORTKEY, consultez CREATE TABLE.

ALTER TRUTKEY AUTO

Clause qui modifie ou ajoute la clé de tri de la table cible à AUTO. ALTER SORTKEY AUTO n'est pas pris en charge pour les tables temporaires.

Lorsque vous modifiez une clé de tri en AUTO, Amazon Redshift conserve la clé de tri existante de la table.

Si Amazon Redshift détermine qu'une nouvelle clé de tri améliorera les performances des requêtes, Amazon Redshift peut modifier la clé de tri de votre table à l'avenir.

Pour plus d'informations sur SORTKEY AUTO, consultez CREATE TABLE.

Pour afficher la clé de tri d'une table, interrogez la vue catalogue système SVV_TABLE_INFO. Pour plus d'informations, consultez SVV_TABLE_INFO. Pour afficher les recommandations Amazon Redshift Advisor pour les tables, recherchez la vue catalogue système SVV_ALTER_TABLE_RECOMMENDATIONS. Pour plus d'informations, consultez SVV_ALTER_TABLE_RECOMMENDATIONS. Pour afficher les actions effectuées par Amazon Redshift, interrogez la vue catalogue système SVL_AUTO_WORKER_ACTION. Pour plus d'informations, consultez SVL_AUTO_WORKER_ACTION.

ALTER SORTKEY NONE

Clause qui supprime la clé de tri de la table cible.

Si la clé de tri était précédemment définie sur AUTO, la table n'est plus candidate à l'optimisation automatique.

ALTER ENCODE AUTO

Clause qui modifie le type d'encodage des colonnes de la table cible en AUTO. Lorsque vous modifiez l'encodage en AUTO, Amazon Redshift conserve le type d'encodage existant des

colonnes de la table. Ensuite, si Amazon Redshift détermine qu'un nouveau type d'encodage peut améliorer les performances de la requête, Amazon Redshift peut modifier le type d'encodage des colonnes de la table.

Si vous modifiez une ou plusieurs colonnes pour spécifier un encodage, Amazon Redshift n'ajuste plus automatiquement l'encodage pour toutes les colonnes de la table. Les colonnes conservent les paramètres d'encodage actuels.

Les actions suivantes n'affectent pas le paramètre ENCODE AUTO pour la table :

- Renommage de la table.
- Modification du paramètre DISTSTYLE ou SORTKEY pour la table.
- Ajout ou suppression d'une colonne avec un paramètre ENCODE.
- Utilisation de l'option COMPUPDATE de la commande COPY. Pour plus d'informations, consultez Opérations de chargement de données.

Pour afficher l'encodage d'une table, interrogez la vue catalogue système SVV_TABLE_INFO. Pour plus d'informations, consultez SVV_TABLE_INFO.

RENAME COLUMN nom_colonne TO nouveau_nom

Clause qui renomme une colonne selon la valeur spécifiée dans nouveau_nom. La longueur maximale d'un nom de colonne est de 127 octets ; les noms plus longs sont tronqués à 127 octets. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

ADD [COLUMN] nom_colonne

Clause qui ajoute une colonne avec le nom spécifié à la table. Vous ne pouvez ajouter qu'une seule colonne à chaque instruction ALTER TABLE.

Vous ne pouvez pas ajouter une colonne qui soit la clé de distribution (DISTKEY) ou une clé de tri (SORTKEY) de la table.

Vous ne pouvez pas utiliser une commande ALTER TABLE ADD COLUMN pour modifier les attributs de table et de colonne suivants :

- UNIQUE
- PRIMARY KEY
- REFERENCES (clé étrangère)
- IDENTITY ou GENERATED BY DEFAULT AS IDENTITY

La longueur maximale d'un nom de colonne est de 127 octets ; les noms plus longs sont tronqués à 127 octets. Le nombre maximal de colonnes que vous pouvez définir dans une seule table est de 1 600.

Les restrictions suivantes s'appliquent lorsque vous ajoutez une colonne à une table externe :

- Vous ne pouvez pas ajouter une colonne à une table externe dont les contraintes de table sont DEFAULT, ENCODE, NOT NULL ou NULL.
- Vous ne pouvez pas ajouter des colonnes à une table externe définie à l'aide du format de fichier AVRO.
- Si les pseudo-colonnes sont activées, le nombre maximal de colonnes que vous pouvez définir dans une seule table externe est 1 598. Si les pseudo-colonnes ne sont pas activées, le nombre maximal de colonnes que vous pouvez définir dans une seule table est 1 600.

Pour plus d'informations, consultez CREATE EXTERNAL TABLE.

type_colonne

Type de données de la colonne ajoutée. Pour les colonnes CHAR et VARCHAR, vous pouvez utiliser le mot-clé MAX au lieu de déclarer une longueur maximale. MAX définit la longueur maximale sur 4 096 octets pour CHAR ou 65 535 octets pour VARCHAR. La taille maximum d'un objet GEOMETRY est de 1 048 447 octets.

Pour obtenir des informations sur les types de données pris en charge par Amazon Redshift, consultez Types de données.

DEFAULT expr_défaut

Clause qui attribue une valeur de données par défaut à la colonne. Le type de données de expr_défaut doit correspondre au type de données de la colonne. La valeur DEFAULT doit être une expression exempte de variable. Les sous-requêtes, les références croisées aux autres colonnes de la table active et les fonctions définies par l'utilisateur ne sont pas autorisées.

expr_défaut est utilisé dans toute opération INSERT qui ne spécifie pas une valeur pour la colonne. Si aucune valeur par défaut n'est spécifiée, la valeur par défaut de la colonne est la valeur null.

Si une opération COPY rencontre un champ null dans une colonne qui a une valeur DEFAULT et une contrainte NOT NULL, la commande COPY insère la valeur de expr_défaut.

DEFAULT n'est pas pris en charge pour les tables externes.

ENCODE encodage

Encodage de compression pour une colonne. Par défaut, Amazon Redshift gère automatiquement l'encodage de compression pour toutes les colonnes d'une table si vous ne spécifiez pas d'encodage de compression pour une colonne de la table ou si vous spécifiez l'option ENCODE AUTO pour la table.

Si vous spécifiez l'encodage de compression pour une colonne de la table ou si vous ne spécifiez pas l'option ENCODE AUTO pour la table, Amazon Redshift attribue automatiquement l'encodage de compression aux colonnes pour lesquelles vous ne spécifiez pas l'encodage de compression comme suit :

- Toutes les colonnes de tables temporaires se voient attribuer une compression RAW par défaut.
- Les colonnes qui sont définies comme des clés de tri se voient attribuer une compression RAW.
- Les colonnes qui sont définies comme des types de données BOOLEAN, REAL, DOUBLE PRECISION ou GEOMETRY ou GEOGRAPHY se voient attribuer une compression RAW.
- Les colonnes définies comme SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ sont soumises à une compression. AZ64
- Les colonnes définies comme CHAR, VARCHAR ou VARBYTE sont affectées à la compression LZO.

> **Note**
>
> Si vous ne souhaitez pas qu'une colonne soit compressée, spécifiez explicitement un encodage RAW (brut).

Les encodages compression encodings (p. 69) suivants sont pris en charge :

- AZ64
- BYTEDICT
- DELTA
- DELTA32 K
- LZO
- MOSTLY8

- MOSTLY16
- MOSTLY32
- RAW (aucune compression)
- RUNLENGTH
- TEXT255
- TEXT32 K
- ZSTD

ENCODE n'est pas pris en charge pour les tables externes.

NOT NULL | NULL

NOT NULL spécifie que la colonne n'est pas autorisée à contenir des valeurs null. NULL, valeur par défaut, spécifie que la colonne accepte les valeurs null.

NOT NULL et NULL ne sont pas pris en charge pour les tables externes.

COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE }

Clause qui spécifie si la recherche de chaîne ou la comparaison sur la colonne est CASE_SENSITIVE (sensible à la casse) ou CASE_INSENSITIVE (insensible à la casse). La valeur par défaut est la même que la configuration actuelle de sensibilité à la casse de la base de données.

Pour rechercher les informations de classement de la base de données, utilisez la commande suivante :

```
SELECT db_collation();

db_collation
----------------
 case_sensitive
(1 row)
```

DROP [COLUMN] nom_colonne

Nom de la colonne à supprimer de la table.

Vous ne pouvez pas supprimer la dernière colonne d'une table. Une table doit avoir au moins une colonne.

Vous ne pouvez pas supprimer une colonne qui est la clé de distribution (DISTKEY) ou une clé de tri (SORTKEY) de la table. Le comportement par défaut pour DROP COLUMN est RESTRICT si la colonne a des objets dépendants, tels qu'une vue, une clé primaire, une clé étrangère ou une restriction UNIQUE.

Les restrictions suivantes s'appliquent lors de la suppression d'une colonne d'une table externe :

- Par ailleurs, vous ne pouvez pas supprimer une colonne d'une table externe si cette colonne est utilisée comme partition.
- Vous ne pouvez pas supprimer une colonne d'une table externe définie à l'aide du format de fichier AVRO.
- RESTRICT et CASCADE sont ignorés pour les tables externes.
- Vous ne pouvez pas supprimer les colonnes de la table de politique référencée dans la définition de la politique à moins de supprimer ou de dissocier la stratégie. Cela s'applique également lorsque l'option CASCADE est spécifiée. Vous pouvez supprimer d'autres colonnes dans le tableau des politiques.

Pour de plus amples informations, veuillez consulter CREATE EXTERNAL TABLE.

RESTRICT

Lorsque RESTRICT est utilisé avec DROP COLUMN, cela signifie que la colonne à supprimer n'est pas supprimée dans les cas suivants :

- Si une vue définie référence la colonne en cours de suppression
- Si une clé étrangère référence la colonne
- Si la colonne fait partie d'une clé en plusieurs parties

RESTRICT ne peut pas être utilisée avec CASCADE.

RESTRICT et CASCADE sont ignorés pour les tables externes.

CASCADE

Lorsqu'il est utilisé avec DROP COLUMN, supprime la colonne spécifiée et tout ce qui dépend de cette colonne. CASCADE ne peut pas être utilisée avec RESTRICT.

RESTRICT et CASCADE sont ignorés pour les tables externes.

Les options suivantes s'appliquent uniquement aux tables externes.

SET LOCATION { 's3://compartiment/dossier/' | 's3://compartiment/fichier_manifeste' }

Chemin menant au dossier Amazon S3 qui contient les fichiers de données ou un fichier manifeste qui contient une liste de chemins d'objets Amazon S3. Les compartiments doivent se trouver dans la même région AWS que le cluster Amazon Redshift. Pour obtenir la liste des AWS régions prises en charge, consultezLimites d'Amazon Redshift Spectrum. Pour plus d'informations sur l'utilisation d'un fichier manifeste, consultez l'option LOCATION dans la référence Paramètres CREATE EXTERNAL TABLE.

SET FILE FORMAT format

Format des fichiers de données externes.

Les formats valides sont les suivants :

- AVRO
- PARQUET
- RCFILE
- SEQUENCEFILE
- TEXTFILE

SET TABLE PROPERTIES ('nom_propriété'='valeur_propriété')

Clause qui définit la définition de table pour des propriétés de table d'une table externe.

> **Note**
>
> Les propriétés de table sont sensibles à la casse.

'numRows'='nombre_lignes'

Propriété qui définit la valeur numRows de la définition de table. Afin de mettre à jour explicitement les statistiques d'une table externe, définissez la propriété numRows pour indiquer la taille de la table. Amazon Redshift n'analyse pas les tables externes pour générer les statistiques de table utilisées par l'optimiseur de requête afin de générer un plan de requête. Si des statistiques de table ne sont pas définies pour une table externe, Amazon Redshift génère un plan d'exécution de requête. Ce plan se base sur une hypothèse selon laquelle les tables externes sont les tables les plus volumineuses et les tables locales les tables les plus petites.

’skip.header.line.count’=’nombre_lignes’

Propriété qui définit le nombre de lignes à ignorer au début de chaque fichier source.

PARTITION (colonne_partition=valeur_partition [, ...] SET LOCATION { ’s3://compartiment/dossier’ | ’s3://compartiment/fichier_manifeste’ }

Clause qui définit un nouvel emplacement pour une ou plusieurs colonnes de partition.

ADD [IF NOT EXISTS] PARTITION (partition_column=partition_value [, ...]) LOCATION { ’s3://bucket/folder’ | ’s3://bucket/manifest_file’ } [, ...]

Clause qui ajoute une ou plusieurs partitions. Vous pouvez spécifier plusieurs clauses PARTITION à l’aide d’une seule instruction ALTER TABLE … ADD.

> **Note**
>
> Si vous utilisez le AWS Glue catalogue, vous pouvez ajouter jusqu'à 100 partitions à l'aide d'une seule instruction ALTER TABLE.

La clause IF NOT EXISTS indique que si la partition spécifiée existe déjà, la commande ne doit effectuer aucun changement. Elle indique également que la commande doit renvoyer un message indiquant que la partition existe, plutôt que s’arrêter avec une erreur. Comme cette clause est utile à l’écriture de scripts, ce script n’échoue pas si ALTER TABLE tente d’ajouter une partition qui existe déjà.

DROP PARTITION (colonne_partition=valeur_partition [, ...])

Clause qui supprime la partition spécifiée. La suppression d’une partition modifie uniquement les métadonnées des tables externes. Ceci n’a aucun impact sur les données sur Amazon S3.

ROW LEVEL SECURITY { ON | OFF } [CONJUNCTION TYPE { AND | OR }] [FOR DATASHARES]

Clause qui active ou désactive la sécurité au niveau des lignes pour une relation.

Lorsque la sécurité au niveau des lignes est activée pour une relation, vous pouvez lire uniquement les lignes auxquelles cette politique vous autorise à accéder. Si aucune politique ne vous accorde l’accès à la relation, vous ne pouvez voir aucune ligne pour la relation. Seuls les super-utilisateurs et les utilisateurs ou les rôles dotés du rôle `sys:secadmin` peuvent définir la clause ROW LEVEL SECURITY. Pour de plus amples informations, veuillez consulter Sécurité au niveau des lignes.

- [CONJUNCTION TYPE { AND | OR }]

  Clause qui vous permet de choisir le type de conjonction de la politique de sécurité au niveau des lignes pour une relation. Lorsque plusieurs politiques de sécurité au niveau des lignes sont associées à une relation, vous pouvez les combiner avec la clause AND ou OR. Par défaut, Amazon Redshift associe les politiques RLS à la clause AND. Les super-utilisateurs, utilisateurs ou rôles disposant du rôle `sys:secadmin` peuvent utiliser cette clause pour définir le type de conjonction de la politique de sécurité au niveau des lignes pour une relation. Pour de plus amples informations, veuillez consulter Association de plusieurs politiques par utilisateur.

- FOR DATASHARES

  Clause qui détermine s'il est possible d'accéder à une relation protégée par RLS sur les unités de partage des données. Par défaut, une relation protégée par RLS n'est pas accessible sur une unité de partage des données. Une commande ALTER TABLE ROW LEVEL SECURITY exécutée avec cette clause affecte uniquement la propriété d'accessibilité de l'unité de partage des données de la relation. La propriété ROW LEVEL SECURITY n'est pas modifiée.

  Si vous rendez accessible une relation protégée par RLS via des unités de partage des données, la relation n'est pas sécurisée au niveau des lignes dans la base de données de l'unité de partage des données côté consommateur. La relation conserve sa propriété RLS du côté producteur.

MASQUAGE {ON | OFF} POUR LES PARTAGES DE DONNÉES

Clause qui détermine si une relation protégée par DDM est accessible via des partages de données. Par défaut, une relation protégée par DDM n'est pas accessible via un partage de données. Si vous rendez accessible une relation protégée par DDM via des partages de données, la relation ne bénéficiera pas de protection de masquage dans la base de données partagée côté consommateur. La relation conserve sa propriété de masquage du côté du producteur. Seuls les superutilisateurs et les utilisateurs ou rôles dotés du `sys:secadmin` rôle peuvent définir la clause MASKING FOR DATASHARES. Pour de plus amples informations, veuillez consulter Masquage dynamique des données.

Exemples

Les exemples suivants montrent comment utiliser la commande ALTER TABLE.

- Exemples ALTER TABLE
- Exemples de modification d'une table externe

- Exemples ALTER TABLE ADD et DROP COLUMN

Exemples ALTER TABLE

Les exemples suivants illustrent l'utilisation de base de la commande ALTER TABLE.

Renommer une table ou une vue

La commande suivante permet de renommer la table USERS en USERS_BKUP :

```
alter table users
rename to users_bkup;
```

Vous pouvez également utiliser ce type de commande pour renommer une vue.

Modifier le propriétaire d'une table ou d'une vue

La commande suivante remplace le propriétaire de la table VENUE par l'utilisateur DWUSER :

```
alter table venue
owner to dwuser;
```

Les commandes suivantes créent une vue, puis modifient son propriétaire :

```
create view vdate as select * from date;
alter table vdate owner to vuser;
```

Renommer une colonne

La commande suivante renomme la colonne VENUESEATS de la table VENUE en VENUESIZE :

```
alter table venue
rename column venueseats to venuesize;
```

Supprimer une contrainte de table

Pour supprimer une contrainte de table, telle qu'une contrainte de clé primaire, de clé étrangère ou unique, commencez par rechercher le nom interne de la contrainte. Ensuite, spécifiez le nom de la

contrainte dans la commande ALTER TABLE. L'exemple suivant recherche les contraintes de la table CATEGORY, puis supprime la clé principale avec le nom `category_pkey`.

```
select constraint_name, constraint_type
from information_schema.table_constraints
where constraint_schema ='public'
and table_name = 'category';

constraint_name | constraint_type
----------------+----------------
category_pkey   | PRIMARY KEY

alter table category
drop constraint category_pkey;
```

Modifier une colonne VARCHAR

Pour conserver le stockage, vous pouvez définir une table initialement avec des colonnes VARCHAR dotées de la taille minimale nécessaire pour vos données actives. Vous pouvez modifier ultérieurement la table afin d'augmenter la taille de la colonne si plus tard vous avez besoin d'accueillir plus de chaînes.

L'exemple suivant augmente la taille de la colonne EVENTNAME à VARCHAR (300).

```
alter table event alter column eventname type varchar(300);
```

Modifier une colonne VARBYTE

Pour économiser de l'espace de stockage, vous pouvez d'abord définir une table avec des colonnes VARBYTE avec la taille minimale requise pour vos besoins actuels en matière de données. Vous pouvez modifier ultérieurement la table afin d'augmenter la taille de la colonne si plus tard vous avez besoin d'accueillir plus de chaînes.

L'exemple suivant augmente la taille de la colonne EVENTNAME à VARBYTE (300).

```
alter table event alter column eventname type varbyte(300);
```

Modifier l'encodage de la compression d'une colonne

Vous pouvez modifier le codage de compression d'une colonne. Vous trouverez ci-dessous une série d'exemples illustrant cette approche. La définition de table pour ces exemples est la suivante.

```
create table t1(c0 int encode lzo, c1 bigint encode zstd, c2 varchar(16) encode lzo, c3
 varchar(32) encode zstd);
```

L'instruction suivante modifie le codage de compression de la colonne c0 du codage LZO au codage. AZ64

```
alter table t1 alter column c0 encode az64;
```

L'instruction suivante modifie le codage de compression de la colonne c1 du codage standard Z au AZ64 codage.

```
alter table t1 alter column c1 encode az64;
```

L’instruction suivante modifie l’encodage de compression pour la colonne c2 de l’encodage LZO à l’encodage Byte-dictionary.

```
alter table t1 alter column c2 encode bytedict;
```

L’instruction suivante modifie l’encodage de compression pour la colonne c3 de l’encodage Zstandard à l’encodage Runlength.

```
alter table t1 alter column c3 encode runlength;
```

Modifier une colonne DISTSTYLE KEY DISTKEY

Les exemples suivants montre comment modifier les colonnes DISTSTYLE et DISTKEY d’une table.

Créez une table avec un style de distribution EVEN. La vue SVV_TABLE_INFO montre que la colonne DISTSTYLE est EVEN.

```
create table inventory(
  inv_date_sk int4 not null ,
  inv_item_sk int4 not null ,
  inv_warehouse_sk int4 not null ,
  inv_quantity_on_hand int4
) diststyle even;

Insert into inventory values(1,1,1,1);

select "table", "diststyle" from svv_table_info;
```

```
   table   |   diststyle
-----------+----------------
 inventory |     EVEN
```

Modifiez la table DISTKEY en `inv_warehouse_sk`. La vue SVV_TABLE_INFO montre la colonne `inv_warehouse_sk` en tant que clé de distribution résultante.

```
alter table inventory alter diststyle key distkey inv_warehouse_sk;

select "table", "diststyle" from svv_table_info;

   table   |       diststyle
-----------+-----------------------
 inventory | KEY(inv_warehouse_sk)
```

Modifiez la table DISTKEY en `inv_item_sk`. La vue SVV_TABLE_INFO montre la colonne `inv_item_sk` en tant que clé de distribution résultante.

```
alter table inventory alter distkey inv_item_sk;

select "table", "diststyle" from svv_table_info;

   table   |       diststyle
-----------+-----------------------
 inventory | KEY(inv_item_sk)
```

Modifier une table en DISTSTYLE ALL

Les exemples suivants montrent comment modifier une table en DISTSTYLE ALL.

Créez une table avec un style de distribution EVEN. La vue SVV_TABLE_INFO montre que la colonne DISTSTYLE est EVEN.

```
create table inventory(
  inv_date_sk int4 not null ,
  inv_item_sk int4 not null ,
  inv_warehouse_sk int4 not null ,
  inv_quantity_on_hand int4
) diststyle even;

Insert into inventory values(1,1,1,1);
```

```
select "table", "diststyle" from svv_table_info;

   table   |   diststyle
-----------+----------------
 inventory |     EVEN
```

Modifiez la table DISTSTYLE en ALL. La vue SVV_TABLE_INFO affiche le DISTSYTLE modifié.

```
alter table inventory alter diststyle all;

select "table", "diststyle" from svv_table_info;

   table   |   diststyle
-----------+----------------
 inventory |     ALL
```

Modifier une table SORTKEY

Vous pouvez modifier une table pour avoir une clé de tri composée ou aucune clé de tri.

Dans la définition de table suivante, la table `t1` est définie par une clé de tri entrelacée.

```
create table t1 (c0 int, c1 int) interleaved sortkey(c0, c1);
```

La commande suivante modifie la table d'une clé de tri entrelacée en une clé de tri composée.

```
alter table t1 alter sortkey(c0, c1);
```

La commande suivante modifie la table pour supprimer la clé de tri entrelacée.

```
alter table t1 alter sortkey none;
```

Dans la définition de table suivante, la table `t1` est définie avec une colonne `c0` comme clé de tri.

```
create table t1 (c0 int, c1 int) sortkey(c0);
```

La commande suivante modifie la table `t1` en clé de tri composée.

```
alter table t1 alter sortkey(c0, c1);
```

Modifier une table en ENCODE AUTO

L'exemple suivant montre comment modifier une table en ENCODE AUTO.

La définition de table pour cet exemple est la suivante. `c0`La colonne est définie avec le type de codage AZ64, et la colonne `c1` est définie avec le type de codage LZO.

```
create table t1(c0 int encode AZ64, c1 varchar encode LZO);
```

Pour cette table, l'instruction suivante modifie l'encodage en AUTO.

```
alter table t1 alter encode auto;
```

L'exemple suivant illustre comment modifier une table pour supprimer le paramètre ENCODE AUTO.

La définition de table pour cet exemple est la suivante. Les colonnes de la table sont définies sans encodage. Dans ce cas, l'encodage est par défaut ENCODE AUTO.

```
create table t2(c0 int, c1 varchar);
```

Pour cette table, l'instruction suivante modifie l'encodage de la colonne c0 en LZO. L'encodage de la table n'est plus défini sur ENCODE AUTO.

```
alter table t2 alter column c0 encode lzo;;
```

Modifier le contrôle de sécurité au niveau des lignes

La commande suivante désactive la politique RLS pour la table :

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY OFF;
```

La commande suivante active RLS pour la table :

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;
```

La commande suivante active RLS pour la table et la rend accessible sur les unités de partage des données :

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;
```

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY FOR DATASHARES OFF;
```

La commande suivante active RLS pour la table et la rend inaccessible sur les unités de partage des données :

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON;
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY FOR DATASHARES ON;
```

La commande suivante active RLS et définit le type de conjonction RLS sur OR pour la table :

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON CONJUNCTION TYPE OR;
```

La commande suivante active RLS et définit le type de conjonction RLS sur AND pour la table :

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON CONJUNCTION TYPE AND;
```

Exemples de modification d'une table externe

Les exemples suivants utilisent un compartiment Amazon S3 situé dans la région USA Est (Virginie du Nord) (`us-east-1`) Région AWS et les exemples de tables créés Exemples pour CREATE TABLE. Pour plus d'informations sur l'utilisation de partitions avec des tables externes, consultezPartitionnement des tables externes Redshift Spectrum.

L'exemple suivant définit la propriété de table numRows pour la table externe SPECTRUM.SALES sur 170 000 lignes.

```
alter table spectrum.sales
set table properties ('numRows'='170000');
```

L'exemple suivant modifie l'emplacement de la table externe SPECTRUM.SALES.

```
alter table spectrum.sales
set location 's3://redshift-downloads/tickit/spectrum/sales/';
```

L'exemple suivant définit le format de la table externe SPECTRUM.SALES sur Parquet.

```
alter table spectrum.sales
set file format parquet;
```

L'exemple suivant ajoute une partition pour la table SPECTRUM.SALES_PART.

```
alter table spectrum.sales_part
add if not exists partition(saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/';
```

L'exemple suivant ajoute trois partitions pour la table SPECTRUM.SALES_PART.

```
alter table spectrum.sales_part add if not exists
partition(saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/'
partition(saledate='2008-02-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/'
partition(saledate='2008-03-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/';
```

L'exemple suivant modifie SPECTRUM.SALES_PART afin de supprimer la partition avec `saledate='2008-01-01''`.

```
alter table spectrum.sales_part
drop partition(saledate='2008-01-01');
```

L'exemple suivant définit un nouveau chemin d'accès Amazon S3 pour la partition avec `saledate='2008-01-01'`.

```
alter table spectrum.sales_part
partition(saledate='2008-01-01')
set location 's3://redshift-downloads/tickit/spectrum/sales_partition/
saledate=2008-01-01/';
```

L'exemple suivant remplace le nom de `sales_date` par `transaction_date`.

```
alter table spectrum.sales rename column sales_date to transaction_date;
```

L'exemple suivant définit le mappage de colonne sur le mappage par position pour une table externe qui utilise le format Optimized Row Columnar (ORC).

```
alter table spectrum.orc_example
set table properties('orc.schema.resolution'='position');
```

L'exemple suivant définit le mappage de colonne sur le mappage par nom pour une table externe qui utilise le format ORC.

```
alter table spectrum.orc_example
set table properties('orc.schema.resolution'='name');
```

Exemples ALTER TABLE ADD et DROP COLUMN

Les exemples suivants montrent comment utiliser ALTER TABLE pour ajouter, puis supprimer une colonne de table de base, ainsi que pour supprimer une colonne avec un objet dépendant.

ADD puis DROP une colonne de base

L'exemple suivant ajoute une colonne FEEDBACK_SCORE autonome à la table USERS. Cette colonne contient simplement un nombre entier et la valeur par défaut de cette colonne est NULL (pas de score de commentaire).

Commencez par interroger la table du catalogue PG_TABLE_DEF pour afficher le schéma de la table USERS :

```
column        | type                   | encoding | distkey | sortkey
--------------+------------------------+----------+---------+--------
userid        | integer                | delta    | true    |       1
username      | character(8)           | lzo      | false   |       0
firstname     | character varying(30)  | text32k  | false   |       0
lastname      | character varying(30)  | text32k  | false   |       0
city          | character varying(30)  | text32k  | false   |       0
state         | character(2)           | bytedict | false   |       0
email         | character varying(100) | lzo      | false   |       0
phone         | character(14)          | lzo      | false   |       0
likesports    | boolean                | none     | false   |       0
liketheatre   | boolean                | none     | false   |       0
likeconcerts  | boolean                | none     | false   |       0
likejazz      | boolean                | none     | false   |       0
likeclassical | boolean                | none     | false   |       0
likeopera     | boolean                | none     | false   |       0
likerock      | boolean                | none     | false   |       0
likevegas     | boolean                | none     | false   |       0
likebroadway  | boolean                | none     | false   |       0
likemusicals  | boolean                | none     | false   |       0
```

Maintenant, ajoutez la colonne feedback_score :

```
alter table users
add column feedback_score int
default NULL;
```

Sélectionnez la colonne FEEDBACK_SCORE de la table USERS pour vérifier qu'elle a été ajoutée :

```
select feedback_score from users limit 5;

feedback_score
---------------
NULL
NULL
NULL
NULL
NULL
```

Supprimez la colonne pour rétablir le DDL d'origine :

```
alter table users drop column feedback_score;
```

Suppression d'une colonne avec un objet dépendant

L'exemple suivant supprime une colonne qui a un objet dépendant. En conséquence, l'objet dépendant est également supprimé.

Pour commencer, ajoutez à nouveau la colonne FEEDBACK_SCORE à la table USERS :

```
alter table users
add column feedback_score int
default NULL;
```

Ensuite, créez une vue de la table USERS appelée USERS_VIEW :

```
create view users_view as select * from users;
```

Maintenant, essayez de supprimer la colonne FEEDBACK_SCORE de la table USERS. Cette instruction DROP utilise le comportement par défaut (RESTRICT) :

```
alter table users drop column feedback_score;
```

Amazon Redshift affiche un message d'erreur indiquant que la colonne ne peut pas être supprimée, car un autre objet en dépend.

Essayez à nouveau de supprimer la colonne FEEDBACK_SCORE, cette fois en spécifiant CASCADE pour supprimer tous les objets dépendants :

```
alter table users
drop column feedback_score cascade;
```

ALTER TABLE APPEND

Ajoute des lignes à une table cible en déplaçant les données à partir d'une table source existante. Les données de la table source sont déplacées vers les colonnes correspondantes de la table cible. L'ordre des colonnes n'importe pas. Une fois que les données ont été correctement ajoutées à la table cible, la table source est vide. ALTER TABLE APPEND est généralement beaucoup plus rapide qu'une opération CREATE TABLE AS ou INSERT INTO semblable, car les données sont déplacées, pas dupliquées.

> **Note**
>
> ALTER TABLE APPEND déplace des blocs de données entre la table source et la table cible. Pour améliorer les performances, ALTER TABLE APPEND ne condense pas le stockage dans le cadre de l'opération d'ajout. Par conséquent, le stockage utilisé augmente provisoirement. Pour récupérer de l'espace, exécutez une opération VACUUM.

Les colonnes ayant le même nom doivent aussi avoir des attributs de colonne identiques. Si la table source ou la table cible contient des colonnes qui n'existent pas dans l'autre table, utilisez les paramètres IGNOREEXTRA ou FILLTARGET pour spécifier comment les colonnes supplémentaires doivent être gérées.

Vous ne pouvez pas ajouter une colonne d'identité. Si les deux tables incluent une colonne d'identité, la commande échoue. Si une seule table dispose d'une colonne d'identité, incluez le paramètre FILLTARGET ou IGNOREEXTRA. Pour plus d'informations, consultez Notes d'utilisation d'ALTER TABLE APPEND.

Vous pouvez ajouter une colonne GENERATED BY DEFAULT AS IDENTITY. Vous pouvez mettre à jour des colonnes définies comme GENERATED BY DEFAULT AS IDENTITY avec des valeurs

que vous fournissez. Pour de plus amples informations, veuillez consulter Notes d'utilisation d'ALTER TABLE APPEND.

La table cible doit être une table permanente. Toutefois, la source peut être une table permanente ou une vue matérialisée configurée pour l'ingestion en streaming. Les deux objets doivent utiliser les mêmes style de distribution et clé de distribution, si l'un ou l'autre a été défini. Si les objets sont triés, les deux objets doivent utiliser le même style de tri et définir les mêmes colonnes comme clés de tri.

Une commande ALTER TABLE APPEND valide automatiquement aussitôt l'opération terminée. Elle ne peut pas être annulée. Vous ne pouvez pas exécuter ALTER TABLE APPEND au sein d'un bloc de transaction (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

Privilèges requis

Selon la commande ALTER TABLE APPEND, l'un des privilèges suivants est requis :

- Superuser
- Utilisateurs disposant du privilège système ALTER TABLE
- Utilisateurs disposant des privilèges DELETE et SELECT sur la table source, et des privilèges INSERT sur la table cible.

Syntaxe

```
ALTER TABLE target_table_name APPEND FROM [ source_table_name
 | source_materialized_view_name ]
[ IGNOREEXTRA | FILLTARGET ]
```

L'ajout à partir d'une vue matérialisée fonctionne uniquement dans le cas où votre vue matérialisée est configurée pour Diffusion de l'ingestion vers une vue matérialisée.

Paramètres

nom_table_cible

Nom de la table à laquelle les lignes sont ajoutées. Spécifiez simplement le nom de la table ou choisissez le format nom_schéma.nom_table pour utiliser un schéma spécifique. La table cible doit être une table permanente existante.

FROM nom_table_source

Nom de la table qui fournit les lignes à ajouter. Spécifiez simplement le nom de la table ou choisissez le format nom_schéma.nom_table pour utiliser un schéma spécifique. La table source doit être une table permanente existante.

FROM source_materialized_view_name

Nom de la vue matérialisée qui fournit les lignes à ajouter. L'ajout à partir d'une vue matérialisée fonctionne uniquement dans le cas où votre vue matérialisée est configurée pour Diffusion de l'ingestion vers une vue matérialisée. La vue matérialisée source doit déjà exister.

IGNOREEXTRA

Mot-clé qui spécifie que si la table source inclut des colonnes qui ne sont pas présentes dans la table cible, les données des colonnes supplémentaires doivent être ignorées. Vous ne pouvez pas utiliser IGNOREEXTRA avec FILLTARGET.

FILLTARGET

Mot-clé qui spécifie que si la table cible inclut des colonnes qui ne sont pas présentes dans la table source, les colonnes doivent être remplies avec la valeur de colonne DEFAULT, s'il en a été défini une, ou avec la valeur NULL. Vous ne pouvez pas utiliser IGNOREEXTRA avec FILLTARGET.

Notes d'utilisation d'ALTER TABLE APPEND

ALTER TABLE APPEND déplace uniquement les colonnes identiques de la table source vers la table cible. L'ordre des colonnes n'importe pas.

Si la table source ou la table cible contient des colonnes supplémentaires, utilisez FILLTARGET ou IGNOREEXTRA selon les règles suivantes :

- Si la table source contient des colonnes qui n'existent pas dans la table cible, incluez IGNOREEXTRA. La commande ignore les colonnes supplémentaires de la table source.
- Si la table cible contient des colonnes qui n'existent pas dans la table source, incluez FILLTARGET. La commande remplit les colonnes supplémentaires de la table cible avec la valeur de colonne par défaut ou la valeur IDENTITY, s'il en a été défini une, ou la valeur NULL.
- Si la table source et la table cible contiennent des colonnes supplémentaires, la commande échoue. Vous ne pouvez pas utiliser FILLTARGET et IGNOREEXTRA.

Si une colonne ayant le même nom, mais des attributs différents, existe dans les deux tables, la commande échoue. Les colonnes aux noms similaires doivent avoir en commun les attributs suivants :

- Type de données
- Taille de colonne
- Encodage de compression
- Non null
- Style de tri
- Colonnes de clé de tri
- Style de distribution
- Colonnes de clé de distribution

Vous ne pouvez pas ajouter une colonne d'identité. Si la table source et la table cible possèdent des colonnes d'identité, la commande échoue. Si seule la table source contient une colonne d'identité, incluez le paramètre IGNOREEXTRA afin que la colonne d'identité soit ignorée. Si seule la table cible comporte une colonne d'identité, incluez le paramètre FILLTARGET afin que la colonne d'identité soit renseignée selon la clause IDENTITY définie pour la table. Pour plus d'informations, consultez DEFAULT.

Vous pouvez ajouter une colonne d'identité par défaut avec l'instruction ALTER TABLE APPEND. Pour plus d'informations, consultez CREATE TABLE.

Exemples ALTER TABLE APPEND

Supposons que votre organisation gère une table, SALES_MONTHLY, pour capturer les transactions commerciales actuelles. Vous voulez, chaque mois, déplacer les données de la table des transactions vers la table SALES.

Vous pouvez utiliser les commandes INSERT INTO et TRUNCATE suivantes pour accomplir la tâche.

```
insert into sales (select * from sales_monthly);
truncate sales_monthly;
```

Cependant, vous pouvez effectuer la même opération bien plus efficacement en utilisant une commande ALTER TABLE APPEND.

D'abord, interrogez la table catalogue système PG_TABLE_DEF pour vérifier que les deux tables ont les mêmes colonnes avec des attributs de colonne identiques.

```
select trim(tablename) as table, "column", trim(type) as type,
encoding, distkey, sortkey, "notnull"
from pg_table_def where tablename like 'sales%';

table      | column     | type                          | encoding | distkey | sortkey |
 notnull
-----------+------------+-------------------------------+----------+---------+---------
+--------
sales      | salesid    | integer                       | lzo      | false   |       0 |
 true
sales      | listid     | integer                       | none     | true    |       1 |
 true
sales      | sellerid   | integer                       | none     | false   |       2 |
 true
sales      | buyerid    | integer                       | lzo      | false   |       0 |
 true
sales      | eventid    | integer                       | mostly16 | false   |       0 |
 true
sales      | dateid     | smallint                      | lzo      | false   |       0 |
 true
sales      | qtysold    | smallint                      | mostly8  | false   |       0 |
 true
sales      | pricepaid  | numeric(8,2)                  | delta32k | false   |       0 |
 false
sales      | commission | numeric(8,2)                  | delta32k | false   |       0 |
 false
sales      | saletime   | timestamp without time zone | lzo      | false   |       0 |
 false
salesmonth | salesid    | integer                       | lzo      | false   |       0 |
 true
salesmonth | listid     | integer                       | none     | true    |       1 |
 true
salesmonth | sellerid   | integer                       | none     | false   |       2 |
 true
salesmonth | buyerid    | integer                       | lzo      | false   |       0 |
 true
salesmonth | eventid    | integer                       | mostly16 | false   |       0 |
 true
salesmonth | dateid     | smallint                      | lzo      | false   |       0 |
 true
```

```
salesmonth | qtysold    | smallint                    | mostly8  | false   |       0 |
 true
salesmonth | pricepaid  | numeric(8,2)                | delta32k | false   |       0 |
 false
salesmonth | commission | numeric(8,2)                | delta32k | false   |       0 |
 false
salesmonth | saletime   | timestamp without time zone | lzo      | false   |       0 |
 false
```

Ensuite, regardez la taille de chaque table.

```
select count(*) from sales_monthly;
 count
-------
  2000
(1 row)

select count(*) from sales;
 count
-------
 412,214
(1 row)
```

Maintenant, exécutez la commande ALTER TABLE APPEND suivante.

```
alter table sales append from sales_monthly;
```

Regardez à nouveau la taille de chaque table. La table SALES_MONTHLY a maintenant 0 ligne et la table SALES a augmenté de 2000 lignes.

```
select count(*) from sales_monthly;
 count
-------
     0
(1 row)

select count(*) from sales;
 count
-------
 414214
(1 row)
```

Si la table source a plus de colonnes que la table cible, spécifiez le paramètre IGNOREEXTRA. L'exemple suivant utilise le paramètre IGNOREEXTRA pour ignorer les colonnes supplémentaires de la table SALES_LISTING lors de l'ajout à la table SALES.

```
alter table sales append from sales_listing ignoreextra;
```

Si la table cible a plus de colonnes que la table source, spécifiez le paramètre FILLTARGET. L'exemple suivant utilise le paramètre FILLTARGET pour remplir les colonnes de la table SALES_REPORT qui n'existent pas dans la table SALES_MONTH.

```
alter table sales_report append from sales_month filltarget;
```

L'exemple suivant montre comment utiliser ALTER TABLE APPEND avec une vue matérialisée en tant que source.

```
ALTER TABLE target_tbl APPEND FROM my_streaming_materialized_view;
```

Les noms de table et de vue matérialisée de cet exemple sont des exemples. L'ajout à partir d'une vue matérialisée fonctionne uniquement dans le cas où votre vue matérialisée est configurée pour Diffusion de l'ingestion vers une vue matérialisée. Cela déplace tous les enregistrements de la vue matérialisée source vers une table cible avec le même schéma que la vue matérialisée et laisse la vue matérialisée intacte. Il s'agit du même comportement que lorsque la source des données est une table.

ALTER USER

Modifie un utilisateur de base de données.

Privilèges requis

Les privilèges suivants sont requis pour ALTER USER :

- Superuser
- Utilisateurs disposant du privilège ALTER USER
- Utilisateur actuel qui souhaite modifier son propre mot de passe

Syntaxe

```
ALTER USER username [ WITH ] option [, ... ]

where option is

CREATEDB | NOCREATEDB
| CREATEUSER | NOCREATEUSER
| SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }
| PASSWORD { 'password' | 'md5hash' | 'sha256hash' | DISABLE }
[ VALID UNTIL 'expiration_date' ]
| RENAME TO new_name |
| CONNECTION LIMIT { limit | UNLIMITED }
| SESSION TIMEOUT limit | RESET SESSION TIMEOUT
| SET parameter { TO | = } { value | DEFAULT }
| RESET parameter
| EXTERNALID external_id
```

Paramètres

nom d’utilisateur

Nom de l’utilisateur.

WITH

Mot-clé facultatif.

CREATEDB | NOCREATEDB

L’option CREATEDB permet à l’utilisateur de créer de nouvelles bases de données. NOCREATEDB est la valeur par défaut.

CREATEUSER | NOCREATEUSER

L’option CREATEUSER crée un super-utilisateur avec tous les privilèges de base de données, y compris CREATE USER. La valeur par défaut est NOCREATEUSER. Pour plus d'informations, consultez superuser.

SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }

Clause qui spécifie le niveau d’accès de l’utilisateur sur les tableaux système et les vues Amazon Redshift.

Les utilisateurs ordinaires qui disposent de l'autorisation SYSLOG ACCESS RESTRICTED ne peuvent voir que les lignes générées par cet utilisateur dans les tables et les vues système visibles par l'utilisateur. La valeur par défaut est RESTRICTED.

Les utilisateurs ordinaires qui disposent de l'autorisation SYSLOG ACCESS UNRESTRICTED peuvent voir toutes les lignes des tables et des vues système visibles par l'utilisateur, y compris les lignes générées par un autre utilisateur. UNRESTRICTED ne permet pas à un utilisateur standard d'avoir accès aux tableaux visibles du super-utilisateur. Seuls les super-utilisateurs peuvent afficher les données des tableaux visibles des super-utilisateurs.

> **Note**
>
> Accorder à un utilisateur un accès illimité aux tableaux système permet à celui-ci de voir les données générées par d'autres utilisateurs. Par exemple, STL_QUERY et STL_QUERYTEXT contiennent le texte complet des instructions INSERT, UPDATE et DELETE, qui peuvent contenir des données confidentielles générées par l'utilisateur.

Toutes les lignes de SVV_TRANSACTIONS sont visibles de tous les utilisateurs.

Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

PASSWORD { 'password' | 'md5hash' | 'sha256hash' | DISABLE }

Définit le mot de passe de l'utilisateur.

Par défaut, les utilisateurs peuvent modifier leurs propres mots de passe, sauf si le mot de passe est désactivé. Pour désactiver le mot de passe d'un utilisateur, spécifiez DISABLE. Lorsque le mot de passe d'un utilisateur est désactivé, le mot de passe est supprimé du système et l'utilisateur ne peut se connecter qu'à l'aide d'informations d'identification utilisateur temporaires AWS Identity and Access Management (IAM). Pour plus d'informations, consultez Utilisation de l'authentification IAM pour générer des informations d'identification de l'utilisateur de base de données. Seul un super-utilisateur peut activer ou désactiver des mots de passe. Vous ne pouvez pas désactiver le mot de passe d'un super-utilisateur. Pour activer un mot de passe, exécutez ALTER USER et spécifiez un mot de passe.

Pour en savoir plus sur l'utilisation du paramètre PASSWORD, consultez CREATE USER.

VALID UNTIL 'date_expiration'

Spécifie que le mot de passe a une date d'expiration. Utilisez la valeur `'infinity'` pour éviter d'avoir une date d'expiration. Le type de données valide pour ce paramètre est timestamp.

Seuls les super-utilisateurs peuvent utiliser ce paramètre.

RENAME TO

Renomme l'utilisateur.

nouveau_nom

Nouveau nom de l'utilisateur. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

> **Important**
>
> Lorsque vous renommez un utilisateur, vous devez aussi modifier son mot de passe. Le mot de passe de réinitialisation ne doit pas nécessairement être différent du mot de passe précédent. Comme le nom d'utilisateur est utilisé dans le cadre du chiffrement du mot de passe, quand un utilisateur est renommé, le mot de passe est effacé. L'utilisateur ne sera pas en mesure de se connecter tant que le mot de passe n'aura pas été réinitialisé. Par exemple :
>
> ```
> alter user newuser password 'EXAMPLENewPassword11';
> ```

CONNECTION LIMIT { limite | UNLIMITED }

Le nombre maximum de connexions à la base de données que l'utilisateur est autorisé à ouvrir simultanément. La limite se s'applique pas aux super-utilisateurs. Utilisez le mot-clé UNLIMITED pour autoriser le nombre maximum de connexions simultanées. Une limite sur le nombre de connexions pour chaque base de données peut également s'appliquer. Pour plus d'informations, consultez CREATE DATABASE. La valeur par défaut est UNLIMITED. Pour afficher les connexions en cours, interrogez la vue système STV_SESSIONS.

> **Note**
>
> Si les deux limites de connexion (utilisateurs et base de données) s'appliquent, un emplacement de connexion inutilisé situé entre les deux limites doit également être disponible lorsqu'un utilisateur tente de se connecter.

SESSION TIMEOUT limit | RESET SESSION TIMEOUT

Durée maximale en secondes pendant laquelle une séance reste inactive ou en veille. La plage est comprise entre 60 secondes (une minute) et 1 728 000 secondes (20 jours). Si aucun délai d'expiration de séance n'est défini pour l'utilisateur, le paramètre de cluster s'applique. Pour plus d'informations, consultez Quotas et limites dans Amazon Redshift dans le Guide de gestion Amazon Redshift.

Lorsque vous définissez le délai d'expiration de séance, il s'applique uniquement aux nouvelles séances.

Pour afficher des informations sur les séances utilisateur actives, y compris l'heure de début, le nom d'utilisateur et le délai d'expiration de la séance, interrogez la vue système STV_SESSIONS. Pour afficher des informations sur l'historique des séances utilisateur, interrogez la vue STL_SESSIONS. Pour récupérer des informations sur les utilisateurs de la base de données, y compris les valeurs de délai d'expiration de séance, interrogez la vue SVL_USER_INFO.

SET

Définit un paramètre de configuration avec une nouvelle valeur par défaut pour toutes les séances exécutées par l'utilisateur spécifié.

RESET

Réinitialise un paramètre de configuration à la valeur par défaut d'origine de l'utilisateur spécifié.

paramètre

Nom du paramètre à définir ou à réinitialiser.

valeur

Nouvelle valeur du paramètre.

DEFAULT

Définit le paramètre de configuration avec la valeur par défaut pour toutes les séances exécutées par l'utilisateur spécifié.

EXTERNALID external_id

Identificateur de l'utilisateur associé à un fournisseur d'identité. Le mot de passe de l'utilisateur doit être désactivé. Pour plus d'informations, consultez Fédération de fournisseur d'identité natif pour Amazon Redshift.

Notes d'utilisation

- Tentative de modification de rdsdb : vous ne pouvez pas modifier l'utilisateur nommé `rdsdb`.
- Création d'un mot de passe inconnu — Lorsque vous utilisez l'authentification AWS Identity and Access Management (IAM) pour créer des informations d'identification d'utilisateur de base de données, vous souhaiterez peut-être créer un superutilisateur capable de se connecter uniquement à l'aide d'informations d'identification temporaires. Vous ne pouvez pas désactiver le mot de passe d'un superutilisateur, mais vous pouvez créer un mot de passe inconnu à l'aide d'une chaîne de MD5 hachage générée aléatoirement.

```
alter user iam_superuser password 'md5123456789012345678012345678901 2';
```

- Définition de search_path : lorsque vous définissez le paramètre search_path avec la commande ALTER USER, la modification prend effet à la prochaine connexion de l'utilisateur spécifié. Si vous voulez modifier la valeur search_path de l'utilisateur et de la séance en cours, utilisez une commande SET.
- Définition du fuseau horaire : lorsque vous utilisez SET TIMEZONE avec la commande ALTER USER, la modification prend effet à la prochaine connexion de l'utilisateur spécifié.
- Utilisation de politiques de masquage dynamique des données et de sécurité au niveau des lignes : lorsque votre cluster provisionné ou votre espace de noms sans serveur est soumis à des politiques de masquage dynamique des données ou de sécurité au niveau des lignes, les commandes suivantes sont bloquées pour les utilisateurs standard :

```
ALTER <current_user> SET enable_case_sensitive_super_attribute/
enable_case_sensitive_identifier/downcase_delimited_identifier
```

Seuls les super-utilisateurs et les utilisateurs disposant du privilège ALTER USER peuvent définir ces options de configuration. Pour plus d'informations sur la sécurité au niveau des lignes, consultez Sécurité au niveau des lignes. Pour plus d'informations sur le masquage dynamique des données, consultez Masquage dynamique des données.

Exemples

L'exemple suivant donne à l'utilisateur ADMIN le privilège de créer des bases de données :

```
alter user admin createdb;
```

L'exemple suivant définit le mot de passe de l'utilisateur ADMIN avec la valeur `adminPass9` et définit une heure et une date d'expiration pour le mot de passe :

```
alter user admin password 'adminPass9'
valid until '2017-12-31 23:59';
```

L'exemple suivant renomme l'utilisateur ADMIN en SYSADMIN :

```
alter user admin rename to sysadmin;
```

L'exemple suivant met à jour le délai d'expiration de la séance en veille pour un utilisateur à 300 secondes.

```
ALTER USER dbuser SESSION TIMEOUT 300;
```

Réinitialise le délai d'expiration de la séance en veille de l'utilisateur. Lorsque vous le réinitialisez, le paramètre de cluster s'applique. Vous devez être un super-utilisateur de la base de données pour exécuter cette commande. Pour plus d'informations, consultez Quotas et limites dans Amazon Redshift dans le Guide de gestion Amazon Redshift.

```
ALTER USER dbuser RESET SESSION TIMEOUT;
```

L'exemple suivant met à jour l'ID externe d'un utilisateur nommé bob. Cet espace de noms est `myco_aad`. Si l'espace de noms n'est pas associé à un fournisseur d'identité enregistré, il en résulte une erreur.

```
ALTER USER myco_aad:bob EXTERNALID "ABC123" PASSWORD DISABLE;
```

L’exemple suivant définit le fuseau horaire pour toutes les sessions exécutées par un utilisateur de base de données spécifique. Il modifie le fuseau horaire pour les sessions ultérieures, mais pas pour celle en cours.

```
ALTER USER odie SET TIMEZONE TO 'Europe/Zurich';
```

L'exemple suivant définit le nombre maximal de connexions à la base de données que l'utilisateur bob est autorisé à ouvrir.

```
ALTER USER bob CONNECTION LIMIT 10;
```

ANALYSE

Met à jour les statistiques de table pour une utilisation par le planificateur de requête.

Privilèges requis

Les privilèges suivants sont requis pour ANALYZE :

- Superuser
- Utilisateurs disposant du privilège ANALYZE
- Propriétaire de la relation
- Propriétaire de la base de données avec qui la table est partagée

Syntaxe

```
ANALYZE [ VERBOSE ]
[ [ table_name [ ( column_name [, ...] ) ] ]
[ PREDICATE COLUMNS | ALL  COLUMNS ]
```

Paramètres

VERBOSE

Clause qui renvoie des messages d’information d’avancement relatifs à l’opération ANALYZE. Cette option est utile lorsque vous ne spécifiez pas une table.

table_name

Vous pouvez analyser des tables spécifiques, y compris les tables temporaires. Vous pouvez qualifier la table avec son nom de schéma. Vous pouvez le cas échéant spécifier un nom_table pour analyser une seule table. Vous ne pouvez pas spécifier plus d'un nom_table avec une seule instruction ANALYZE nom_table. Si vous ne spécifiez pas un table_name, toutes les tables de la base de données actuellement connectée sont analysées, y compris les tables persistantes du catalogue système. Amazon Redshift ignore l'analyse d'une table si le pourcentage de lignes qui ont été modifiées depuis la dernière opération ANALYZE est inférieur au seuil d'analyse. Pour plus d'informations, consultez Seuil d'analyse.

Vous n'avez pas besoin d'analyser les tables système Amazon Redshift (tables STL et STV).

column_name

Si vous spécifiez un nom_table, vous pouvez également spécifier une ou plusieurs colonnes de la table (au forme de liste de colonnes séparées entre parenthèses). Si une liste de colonnes est spécifiée, seules les colonnes de la liste sont analysées.

PREDICATE COLUMNS | ALL COLUMNS

Clauses qui indiquent si ANALYZE doit inclure uniquement les colonnes de prédicat. Spécifiez PREDICATE COLUMNS pour analyser uniquement les colonnes qui ont été utilisées comme prédicats dans des requêtes précédentes ou qui sont susceptibles d'être utilisées comme prédicats. Spécifiez ALL COLUMNS pour analyser toutes les colonnes. La valeur par défaut est ALL COLUMNS.

Une colonne est incluse dans l'ensemble de colonnes de prédicat si l'une des conditions suivantes est vraie :

- La colonne a été utilisée dans une requête dans le cadre d'un filtre, d'une condition de jointure ou d'une clause group by.
- La colonne est une clé de distribution.
- La colonne fait partie d'une clé de tri.

Si aucune colonne n'est marquée comme colonne de prédicat, par exemple, parce que la table n'a pas encore été interrogée, toutes les colonnes sont analysées même si PREDICATE COLUMNS est spécifié. Dans ce cas, Amazon Redshift peut répondre par un message tel que Aucune colonne de prédicat trouvée pour « ». *table-name* Analyse de toutes les colonnes. Pour plus d'informations sur les colonnes de prédicat, consultez Analyse des tables.

Notes d'utilisation

Amazon Redshift exécute automatiquement les tables que vous créez avec les commandes suivantes :

- CREATE TABLE AS
- CREATE TEMP TABLE AS
- SELECT INTO

Vous ne pouvez pas analyser une table externe.

Vous n'avez pas besoin d'exécuter la commande ANALYZE sur ces tables lorsqu'elles sont créées initialement. Si vous les modifiez, vous devez les analyser de la même manière que les autres tables.

Seuil d'analyse

Pour réduire le temps de traitement et améliorer les performances globales du système, Amazon Redshift ignore ANALYZE pour une table si le pourcentage de lignes qui ont été modifiées depuis l'exécution de la dernière opération ANALYZE est inférieur au seuil d'analyse spécifié par le paramètre analyze_threshold_percent. Par défaut, `analyze_threshold_percent` est 10. Pour modifier `analyze_threshold_percent` de la séance en cours, exécutez la commande SET. L'exemple suivant passe de `analyze_threshold_percent` à 20 %.

```
set analyze_threshold_percent to 20;
```

Pour analyser des tables lorsque seul un petit nombre de lignes a changé, définissez `analyze_threshold_percent` sur un petit nombre arbitraire. Par exemple, si vous définissez `analyze_threshold_percent` sur 0,01, une table avec 100 000 000 lignes n'est pas ignorée si au moins 10 000 lignes ont été modifiées.

```
set analyze_threshold_percent to 0.01;
```

Si ANALYZE ignore une table, car il ne respecte pas le seuil d'analyse, Amazon Redshift renvoie le message suivant.

```
ANALYZE SKIP
```

Pour analyser toutes les tables même si aucune ligne n'a été modifiée, définissez `analyze_threshold_percent` sur 0.

Pour afficher les résultats des opérations ANALYZE, interrogez la table système STL_ANALYZE.

Pour plus d'informations sur l'analyse des tables, consultez Analyse des tables.

Exemples

Analysez toutes les tables de la base de données TICKIT et renvoyez les informations d'avancement.

```
analyze verbose;
```

Analysez la table LISTING uniquement.

```
analyze listing;
```

Analysez les colonnes VENUEID et VENUENAME de la table VENUE.

```
analyze venue(venueid, venuename);
```

Analysez uniquement les colonnes de prédicat de la table VENUE.

```
analyze venue predicate columns;
```

ANALYZE COMPRESSION

Effectue l'analyse de compression et produit un rapport avec l'encodage de suppression suggéré pour les tables analysées. Pour chaque colonne, le rapport inclut une estimation de la réduction potentielle de l'espace disque par rapport au codage RAW.

Syntaxe

```
ANALYZE COMPRESSION
[ [ table_name ]
[ ( column_name [, ...] ) ] ]
[COMPROWS numrows]
```

Paramètres

table_name

Vous pouvez analyser la compression pour des tables spécifiques, y compris les tables temporaires. Vous pouvez qualifier la table avec son nom de schéma. Vous pouvez le cas

échéant spécifier un nom_table pour analyser une seule table. Si vous ne spécifiez pas un nom_table, toutes les tables de la base de données actuellement connectée sont analysées. Vous ne pouvez pas spécifier plus d'un nom_table avec une seule instruction ANALYZE COMPRESSION.

column_name

Si vous spécifiez un nom_table, vous pouvez également spécifier une ou plusieurs colonnes de la table (au forme de liste de colonnes séparées entre parenthèses).

COMPROWS

Nombre de lignes à utiliser comme taille de l'échantillon pour l'analyse de la compression. L'analyse est exécutée sur les lignes de chaque tranche de données. Par exemple, si vous spécifiez COMPROWS 1000000 (1 000 000) et que le système contient en tout 4 tranches, pas plus de 250 000 lignes par tranche ne sont lues et analysées. Si COMPROWS n'est pas spécifié, la taille de l'échantillon par défaut est de 100 000 par tranche. Les valeurs COMPROWS inférieures à la valeur par défaut de 100 000 lignes par tranche sont automatiquement mises à niveau avec la valeur par défaut. Cependant, l'analyse de compression ne génère pas de recommandations si la quantité de données dans la table n'est pas suffisante pour produire un échantillon significatif. Si le nombre COMPROWS est supérieur au nombre de lignes de la table, la commande ANALYZE COMPRESSION continue de se poursuivre et exécute l'analyse de la compression sur toutes les lignes disponibles. L'utilisation de COMPROWS entraîne une erreur si aucune table n'est spécifiée.

numrows

Nombre de lignes à utiliser comme taille de l'échantillon pour l'analyse de la compression. La plage acceptée pour numrows est un nombre compris entre 1000 et 1000000000 (1 000 000 000).

Notes d'utilisation

ANALYSER COMPRESSION acquiert un verrou de table exclusif, qui empêche les lectures et les écritures simultanées sur la table. Exécutez uniquement la commande COMPRESSION lorsque la table est inactive.

Exécutez ANALYZE COMPRESSION afin d'obtenir des recommandations pour les schémas d'encodage de colonne, en fonction d'un échantillon du contenu de la table. ANALYZE COMPRESSION est un outil consultatif et ne modifie pas les encodages de colonne de la table. Vous pouvez appliquer l'encodage suggéré en recréant la table ou en créant une nouvelle table

avec le même schéma. La recréation d'une table non compressée à l'aide de schémas de codage appropriés peut réduire considérablement son encombrement sur le disque. Cette approche permet d'économiser de l'espace disque et d'améliorer les performances des requêtes pour les charges de travail liées aux I/O.

ANALYZE COMPRESSION ignore la phase d'analyse réelle et renvoie directement le type d'encodage d'origine sur n'importe quelle colonne désignée comme SORTKEY. En effet, les analyses limitées à la plage peuvent fonctionner de façon médiocre lorsque les colonnes SORTKEY sont compressées beaucoup plus fortement que les autres colonnes.

Exemples

L'exemple suivant affiche l'encodage et le pourcentage estimé de réduction pour les colonnes de la table LISTING uniquement :

```
analyze compression listing;

  Table  |     Column     | Encoding | Est_reduction_pct
---------+----------------+----------+-------------------
 listing | listid         | az64     | 40.96
 listing | sellerid       | az64     | 46.92
 listing | eventid        | az64     | 53.37
 listing | dateid         | raw      | 0.00
 listing | numtickets     | az64     | 65.66
 listing | priceperticket | az64     | 72.94
 listing | totalprice     | az64     | 68.05
 listing | listtime       | az64     | 49.74
```

L'exemple suivant analyse les colonnes QTYSOLD, COMMISSION et SALETIME de la table SALES.

```
analyze compression sales(qtysold, commission, saletime);

 Table |   Column   | Encoding | Est_reduction_pct
-------+------------+----------+-------------------
 sales | salesid    | N/A      | 0.00
 sales | listid     | N/A      | 0.00
 sales | sellerid   | N/A      | 0.00
 sales | buyerid    | N/A      | 0.00
 sales | eventid    | N/A      | 0.00
 sales | dateid     | N/A      | 0.00
 sales | qtysold    | az64     | 83.06
```

```
 sales | pricepaid   | N/A      | 0.00
 sales | commission | az64     | 71.85
 sales | saletime   | az64     | 49.63
```

ATTACH MASKING POLICY

Attache une politique de masquage dynamique des données existante à une colonne. Pour plus d'informations sur le masquage dynamique des données, consultez Masquage dynamique des données.

Les super-utilisateurs et les utilisateurs ou les rôles disposant du rôle sys:secadmin peuvent attacher une politique de masquage.

Syntaxe

```
ATTACH MASKING POLICY policy_name
   ON { relation_name }
   ( {output_columns_names | output_path} ) [ USING ( {input_column_names | input_path
 )} ]
   TO { user_name | ROLE role_name | PUBLIC }
   [ PRIORITY priority ];
```

Paramètres

policy_name

Nom de la politique de masquage à attacher.

relation_name

Nom de la relation à laquelle attacher la politique de masquage.

output_column_names

Noms des colonnes auxquelles la politique de masquage s'appliquera.

output_paths

Chemin complet de l'objet SUPER auquel la politique de masquage s'appliquera, y compris le nom de colonne. Par exemple, pour une relation avec une colonne de type SUPER nommée `person`, output_path peut être `person.name.first_name`.

input_column_names

Noms des colonnes que la politique de masquage prendra comme entrées. Ce paramètre est facultatif. S'il n'est pas spécifié, la politique de masquage utilise output_column_names comme entrées.

input_paths

Chemin complet de l'objet SUPER que la politique de masquage prendra comme entrée. Ce paramètre est facultatif. S'il n'est pas spécifié, la politique de masquage utilise output_path pour les entrées.

user_name

Nom de l'utilisateur auquel la politique de masquage s'attachera. Vous ne pouvez pas attacher deux politiques à la même combinaison d'utilisateur et de colonne, ou de rôle et de colonne. Vous pouvez attacher une politique à un utilisateur et une autre politique au rôle de l'utilisateur. Dans ce cas, la politique ayant la priorité la plus élevée s'applique.

Vous ne pouvez définir qu'un seul user_name, role_name et PUBLIC dans une seule commande ATTACH MASKING POLICY.

role_name

Nom du rôle auquel la politique de masquage s'attachera. Vous ne pouvez pas associer deux politiques à la même column/role paire. Vous pouvez attacher une politique à un utilisateur et une autre politique au rôle de l'utilisateur. Dans ce cas, la politique ayant la priorité la plus élevée s'applique.

Vous ne pouvez définir qu'un seul user_name, role_name et PUBLIC dans une seule commande ATTACH MASKING POLICY.

PUBLIC

Attache la politique de masquage à tous les utilisateurs accédant à la table. Pour qu'elles s'appliquent, vous devez accorder aux autres politiques de masquage attachées à des column/role paires column/user ou à des spécifiques une priorité supérieure à la politique PUBLIQUE.

Vous ne pouvez définir qu'un seul user_name, role_name et PUBLIC dans une seule commande ATTACH MASKING POLICY.

priority

Priorité de la politique de masquage. Lorsque plusieurs politiques de masquage s'appliquent à la requête d'un utilisateur donné, la politique de priorité la plus élevée s'applique.

Vous ne pouvez pas attacher deux politiques différentes à la même colonne avec la même priorité, même si les deux politiques sont attachées à différents utilisateurs ou rôles. Vous pouvez associer la même politique plusieurs fois au même ensemble de paramètres de table, de colonne de sortie, de colonne d'entrée et de priorité, à condition que l'utilisateur ou le rôle auquel la politique est attachée soit différent à chaque fois.

Vous ne pouvez pas appliquer de politique à une colonne ayant la même priorité qu'une autre politique attachée à cette colonne, même si elles concernent des rôles différents. Ce champ est facultatif. Si vous ne spécifiez pas de priorité, la politique de masquage attache par défaut une priorité de 0.

ATTACH RLS POLICY

Attachez une politique de sécurité au niveau des lignes sur une table à un ou plusieurs utilisateurs ou rôles.

Les super-utilisateurs et les utilisateurs ou les rôles qui disposent du rôle `sys:secadmin` peuvent attacher une stratégie.

Syntaxe

```
ATTACH RLS POLICY policy_name ON [TABLE] table_name [, ...]
TO { user_name | ROLE role_name | PUBLIC } [, ...]
```

Paramètres

policy_name

Nom de la politique .

ON [TABLE] table_name [, ...]

Relation à laquelle la politique de sécurité au niveau des lignes est attachée.

TO { user_name | ROLE role_name | PUBLIC} [, ...]

Spécifie si la politique est attachée à un ou plusieurs utilisateurs ou rôles spécifiés.

Notes d'utilisation

Lorsque vous utilisez l'instruction ATTACH RLS POLICY, tenez compte des points suivants :

- La table attachée doit contenir toutes les colonnes répertoriées dans la clause WITH de l'instruction de création de la stratégie.
- Amazon Redshift RLS permet d'associer des politiques RLS aux objets suivants :
  - Tables
  - Vues
  - Vues à liaison tardive
  - Vues matérialisées
- Amazon Redshift RLS ne permet pas d'associer des politiques RLS aux objets suivants :
  - Tables de catalogue
  - Relations entre bases de données
  - Tables externes
  - Tables temporaires
  - Tables de recherche de politiques
  - Tables de base des vues matérialisées
- Les politiques RLS associées aux superutilisateurs ou aux utilisateurs `sys:secadmin` autorisés sont ignorées.

Exemples

L'exemple suivant attache une politique sur une table à un rôle.

```
ATTACH RLS POLICY policy_concerts ON tickit_category_redshift TO ROLE analyst, ROLE
 dbadmin;
```

BEGIN

Démarre une transaction. Synonyme de START TRANSACTION.

Une transaction est une même unité de travail logique qui se compose d'une commande ou de plusieurs commandes. En général, toutes les commandes d'une transaction s'exécutent sur un instantané de la base de données dont l'heure de démarrage est déterminée par la valeur définie pour le paramètre de configuration `transaction_snapshot_begin`.

Par défaut, les opérations Amazon Redshift (requêtes, instructions DDL, charges) sont automatiquement validées sur la base de données. Si vous souhaitez suspendre la validation d'une

opération jusqu'à la fin de la tâche suivante, vous devez ouvrir une transaction avec l'instruction BEGIN, exécuter les commandes nécessaires et fermer la transaction avec une instruction COMMIT ou FIN. Si nécessaire, vous pouvez utiliser une instruction ROLLBACK pour arrêter une transaction en cours. Une exception à ce comportement est la commande TRUNCATE, qui valide la transaction dans laquelle elle est exécutée et ne peut pas être annulée.

Syntaxe

```
BEGIN [ WORK | TRANSACTION ] [ ISOLATION LEVEL option ] [ READ WRITE | READ ONLY ]

START TRANSACTION [ ISOLATION LEVEL option ] [ READ WRITE | READ ONLY ]

Where option is

SERIALIZABLE
| READ UNCOMMITTED
| READ COMMITTED
| REPEATABLE READ

Note: READ UNCOMMITTED, READ COMMITTED, and REPEATABLE READ have no
operational impact and map to SERIALIZABLE in Amazon Redshift. You can see database
 isolation levels on your cluster
by querying the stv_db_isolation_level table.
```

Paramètres

WORK

Mot-clé facultatif.

TRANSACTION

Mot-clé facultatif ; WORK et TRANSACTION sont synonymes.

ISOLATION LEVEL SERIALIZABLE

Comme l'isolement sérialisable est pris en charge par défaut, le comportement de la transaction est le même, que cette syntaxe soit incluse ou pas dans l'instruction. Pour plus d'informations, consultez Gestion des opérations d'écriture simultanées. Aucun autre niveau d'isolement n'est pris en charge.

> **Note**
>
> Le langage SQL standard définit quatre niveaux d'isolement des transactions afin d'éviter les lectures non validées (où une transaction lit les données écrites par une transaction simultanée non validée), les lectures non reproductibles (où une transaction relit les données qu'elle a lues précédemment et découvre que les données ont été modifiées par une autre transaction qui les a validées depuis la lecture initiale), et les lectures fantôme (où une opération ré-exécute une requête, renvoie un ensemble de lignes qui satisfont à une condition de recherche et découvre que l'ensemble des lignes a changé en raison d'une autre transaction récemment validée) :
>
> - Lecture non validée : lectures non validées, lectures non reproductibles et lectures fantôme possibles.
> - Lecture validée : lectures non reproductibles et lectures fantôme possibles.
> - Lecture reproductible : lectures fantôme possibles.
> - Sérialisable : empêche les lectures non validées, les lectures non reproductibles et les lectures fantôme.
>
> Même si vous pouvez utiliser n'importe lequel des quatre niveaux d'isolement des transactions, Amazon Redshift traite tous les niveaux d'isolement comme sérialisables.

READ WRITE

Accorde à la transaction les autorisations en lecture et en écriture.

READ ONLY

Accorde à la transaction les autorisations en lecture seule.

Exemples

L'exemple suivant démarre un bloc de transaction sérialisable :

```
begin;
```

L'exemple suivant démarre le bloc de transaction par un niveau d'isolement sérialisable, et les autorisations en lecture et écriture :

```
begin read write;
```

CALL

Exécute une procédure stockée. La commande CALL doit inclure le nom de la procédure ainsi que les valeurs des arguments en entrée. Vous devez appeler une procédure stockée à l'aide de l'instruction CALL.

> **Note**
>
> CALL ne peut pas faire partie d'une requête classique.

Syntaxe

```
CALL sp_name ( [ argument ] [, ...] )
```

Paramètres

sp_name

Nom de la procédure à exécuter.

argument

Type de valeur de l'argument en entrée. Ce paramètre peut également être un nom de fonction, par exemple `pg_last_query_id()`. Vous pouvez utiliser les requêtes en tant qu'arguments CALL.

Notes d'utilisation

Les procédures stockées Amazon Redshift prennent en charge les appels imbriqués et récursifs, comme décrit ci-dessous. En outre, assurez-vous que l'assistance de votre chauffeur est up-to-date également décrite ci-dessous.

Rubriques

- Appels imbriqués
- Prise en charge du pilote

Appels imbriqués

Les procédures stockées Amazon Redshift prennent en charge les appels imbriqués et récursifs. Le nombre maximal de niveaux d'imbrication autorisé est 16. Les appels imbriqués peuvent encapsuler la logique métier en procédures plus petites, pouvant être partagées par plusieurs appelants.

Si vous appelez une procédure imbriquée comportant des paramètres de sortie, la procédure interne doit définir des arguments INOUT. Dans ce cas, la procédure interne est transmise dans une variable non constante. Les arguments OUT ne sont pas autorisés. Ce comportement se produit car une variable est nécessaire pour contenir la sortie de l'appel interne.

Cette relation entre les procédures internes et externes est enregistrée dans la colonne `from_sp_call` de SVL_STORED_PROC_CALL.

L'exemple suivant illustre la transmission de variables à un appel de procédure imbriquée via des arguments INOUT.

```
CREATE OR REPLACE PROCEDURE inner_proc(INOUT a int, b int, INOUT c int) LANGUAGE
 plpgsql
AS $$
BEGIN
  a := b * a;
  c := b * c;
END;
$$;

CREATE OR REPLACE PROCEDURE outer_proc(multiplier int) LANGUAGE plpgsql
AS $$
DECLARE
  x int := 3;
  y int := 4;
BEGIN
  DROP TABLE IF EXISTS test_tbl;
  CREATE TEMP TABLE test_tbl(a int, b varchar(256));
  CALL inner_proc(x, multiplier, y);
  insert into test_tbl values (x, y::varchar);
END;
$$;

CALL outer_proc(5);

SELECT * from test_tbl;
```

```
 a  | b
----+----
 15 | 20
(1 row)
```

Prise en charge du pilote

Nous vous recommandons de mettre à niveau vos pilotes JDBC (Java Database Connectivity) et OBDC (Open Database Connectivity) avec la dernière version, qui prend en charge les procédures stockées Amazon Redshift.

Vous pourrez peut-être utiliser votre pilote existant si votre outil client utilise les opération d'API transmises au serveur via l'instruction CALL. Les paramètres de sortie, le cas échéant, sont renvoyés sous la forme d'un ensemble de résultats d'une ligne.

Les dernières versions des pilotes JDBC et ODBC Amazon Redshift prennent en charge les métadonnées pour la détection de procédures stockées. Elles offrent également une prise en charge `CallableStatement` pour les applications Java personnalisées. Pour plus d'informations sur les pilotes, consultez Connexion à un cluster Amazon Redshift à l'aide des outils clients SQL dans le Guide de gestion Amazon Redshift.

Les exemples suivants montrent comment utiliser différentes opérations d'API du pilote JDBC pour les appels de procédures stockées.

```
void statement_example(Connection conn) throws SQLException {
  statement.execute("CALL sp_statement_example(1)");
}

void prepared_statement_example(Connection conn) throws SQLException {
  String sql = "CALL sp_prepared_statement_example(42, 84)";
  PreparedStatement pstmt = conn.prepareStatement(sql);
  pstmt.execute();
}

void callable_statement_example(Connection conn) throws SQLException {
  CallableStatement cstmt = conn.prepareCall("CALL sp_create_out_in(?,?)");
  cstmt.registerOutParameter(1, java.sql.Types.INTEGER);
  cstmt.setInt(2, 42);
  cstmt.executeQuery();
  Integer out_value = cstmt.getInt(1);
}
```

Exemples

L'exemple suivant appelle le nom de procédure `test_spl`.

```
call test_sp1(3,'book');
INFO:  Table "tmp_tbl" does not exist and will be skipped
INFO:  min_val = 3, f2 = book
```

L'exemple suivant appelle le nom de procédure `test_spl2`.

```
call test_sp2(2,'2019');

         f2          | column2
---------------------+---------
 2019+2019+2019+2019 | 2
(1 row)
```

ANNULER

Annule une requête de base de données qui s'exécute simultanément.

La commande CANCEL nécessite l'ID de processus ou l'ID de session de la requête en cours d'exécution et affiche un message de confirmation pour vérifier que la requête a été annulée.

Privilèges requis

Les privilèges suivants sont requis pour CANCEL :

- Super-utilisateur annulant sa propre requête
- Super-utilisateur annulant la requête d'un utilisateur
- Utilisateurs disposant du privilège CANCEL et annulant la requête d'un utilisateur
- Utilisateur annulant sa propre requête

Syntaxe

```
CANCEL process_id [ 'message' ]
```

Paramètres

process_id

Pour annuler une requête exécutée dans un cluster Amazon Redshift, utilisez le `pid` (ID de STV_RECENTS processus) correspondant à la requête que vous souhaitez annuler.

Pour annuler une requête exécutée dans un groupe de travail Amazon Redshift Serverless, utilisez le formulaire `session_id` correspondant à la requête SYS_QUERY_HISTORY que vous souhaitez annuler.

’message’

Message de confirmation facultatif qui s’affiche lorsque l’annulation de la requête est terminée. Si vous ne spécifiez pas un message, Amazon Redshift affiche le message par défaut en tant que vérification. Vous devez placer le message entre guillemets simples.

Notes d’utilisation

Vous ne pouvez pas annuler une requête en spécifiant un ID de requête ; vous devez spécifier l'ID de processus (PID) ou l'ID de session de la requête. Vous pouvez uniquement annuler les requêtes exécutées simultanément par votre utilisateur. Les super-utilisateurs peuvent annuler toutes les requêtes.

Si les requêtes de plusieurs séances maintiennent des verrous sur la même table, vous pouvez utiliser la fonction PG_TERMINATE_BACKEND pour mettre fin à l’une des séances. Cela oblige toutes les transactions en cours d’exécution dans la séance terminée à libérer tous les verrous et à restaurer la transaction. Interrogez la table système STV_LOCKS pour afficher les verrous actuellement détenus.

Suite à certains événements internes, Amazon Redshift peut redémarrer une séance active et attribuer un nouveau PID. Si le PID a été modifié, vous pourriez recevoir le message d’erreur suivant :

```
Session <PID> does not exist. The session PID might have changed. Check the
 stl_restarted_sessions system table for details.
```

Pour trouver le nouveau PID, interrogez la table système STL_RESTARTED_SESSIONS et filtrez sur la colonne `oldpid`.

```
select oldpid, newpid from stl_restarted_sessions where oldpid = 1234;
```

Exemples

Pour annuler une requête en cours d'exécution dans un cluster Amazon Redshift, récupérez d'abord l'ID de processus de la requête que vous souhaitez annuler. Pour déterminer le processus IDs de toutes les requêtes en cours d'exécution, tapez la commande suivante :

```
select pid, starttime, duration,
trim(user_name) as user,
trim (query) as querytxt
from stv_recents
where status = 'Running';

pid |         starttime          | duration |   user   |    querytxt
-----+----------------------------+----------+----------+-----------------
802 | 2008-10-14 09:19:03.550885 |      132 | dwuser | select
venuename from venue where venuestate='FL', where venuecity not in
('Miami' , 'Orlando');
834 | 2008-10-14 08:33:49.473585 |  1250414 | dwuser | select *
from listing;
964 | 2008-10-14 08:30:43.290527 |   326179 | dwuser | select
sellerid from sales where qtysold in (8, 10);
```

Vérifiez le texte de la requête pour déterminer quel identifiant de processus (PID) correspond à la requête que vous voulez annuler.

Tapez la commande suivante pour utiliser le PID 802 afin d’annuler cette requête :

```
cancel 802;
```

La séance au cours de laquelle la requête était en cours d’exécution affiche le message suivant :

```
ERROR:  Query (168) cancelled on user's request
```

où 168 est l’ID de la requête (pas l’ID du processus utilisé pour annuler la requête).

Sinon, vous pouvez spécifier un message de confirmation personnalisé à afficher à la place du message par défaut. Pour spécifier un message personnalisé, incluez votre message entre guillemets simples à la fin de la commande CANCEL :

```
cancel 802 'Long-running query';
```

La séance au cours de laquelle la requête était en cours d'exécution affiche le message suivant :

```
ERROR:  Long-running query
```

CLOSE

(Facultatif) Ferme toutes les ressources disponibles associées à un curseur ouvert. Comme COMMIT, FIN et ROLLBACK ferment automatiquement le curseur, il n'est pas nécessaire d'utiliser la commande CLOSE pour fermer explicitement le curseur.

Pour plus d'informations, consultez DECLARE, FETCH.

Syntaxe

```
CLOSE cursor
```

Paramètres

curseur

Nom du curseur à fermer.

Exemple CLOSE

Les commandes suivantes ferment le curseur et effectuent une validation, ce qui met fin à la transaction :

```
close movie_cursor;
commit;
```

COMMENT

Crée ou modifie un commentaire relatif à un objet de base de données.

Syntaxe

```
COMMENT ON
```

```
{
TABLE object_name |
COLUMN object_name.column_name |
CONSTRAINT constraint_name ON table_name |
DATABASE object_name |
VIEW object_name
}
IS 'text' | NULL
```

Paramètres

nom d’objet

Nom de l’objet de base de données auquel s’applique le commentaire. Vous pouvez ajouter un commentaire aux objets suivants :

- TABLE
- COLUMN (accepte également un nom_colonne).
- CONSTRAINT (accepte également un nom_contrainte et un nom_table).
- DATABASE
- VIEW
- SCHEMA (SCHÉMA)

IS ’text’ | NULL

Le texte du commentaire que vous souhaitez ajouter ou remplacer pour l’objet spécifié. La chaîne de texte est de type TEXT. Placez le commentaire entre guillemets simples. Spécifiez la valeur sur NULL pour supprimer le texte du commentaire.

column_name

Nom de la colonne commentée. Paramètre de COLUMN. Suit une table spécifiée dans `object_name`.

nom_contrainte

Nom de la contrainte commentée. Paramètre de CONSTRAINT.

table_name

Nom d’une table contenant la contrainte. Paramètre de CONSTRAINT.

Notes d’utilisation

Vous devez être un superutilisateur ou le propriétaire d’un objet de la base de données pour ajouter ou mettre à jour un commentaire.

Les commentaires sur les bases de données ne peuvent s’appliquer qu’à la base de données en cours. Un message d’avertissement s’affiche si vous essayez de faire un commentaire sur une autre base de données. Le même avertissement s’affiche pour les commentaires sur les bases de données qui n’existent pas.

Les commentaires sur les tables externes, les colonnes externes et les colonnes des vues à reliure tardive ne sont pas pris en charge.

Exemples

L’exemple suivant ajoute un commentaire à la table SALES.

```
COMMENT ON TABLE sales IS 'This table stores tickets sales data';
```

L’exemple suivant affiche le commentaire sur la table SALES.

```
select obj_description('public.sales'::regclass);

obj_description
-----------------------------------
This table stores tickets sales data
```

L’exemple suivant supprime un commentaire de la table SALES.

```
COMMENT ON TABLE sales IS NULL;
```

L’exemple suivant ajoute un commentaire à la colonne EVENTID de la table SALES.

```
COMMENT ON COLUMN sales.eventid IS 'Foreign-key reference to the EVENT table.';
```

L’exemple suivant affiche un commentaire sur la colonne EVENTID (colonne numéro 5) de la table SALES.

```
select col_description( 'public.sales'::regclass, 5::integer );
```

```
col_description
----------------------------------------
Foreign-key reference to the EVENT table.
```

L'exemple suivant ajoute un commentaire descriptif à la table EVENT.

```
comment on table event is 'Contains listings of individual events.';
```

Pour afficher les commentaires, interrogez le catalogue système PG_DESCRIPTION. L'exemple suivant renvoie la description de la table EVENT.

```
select * from pg_catalog.pg_description
where objoid =
(select oid from pg_class where relname = 'event'
and relnamespace =
(select oid from pg_catalog.pg_namespace where nspname = 'public') );

objoid | classoid | objsubid | description
-------+----------+----------+----------------------------------------
116658 |     1259 |        0 | Contains listings of individual events.
```

COMMIT

Valide la transaction actuelle sur la base de données. Cette commande rend permanentes les mises à jour de base de données de la transaction.

Syntaxe

```
COMMIT [ WORK | TRANSACTION ]
```

Paramètres

WORK

Mot-clé facultatif. Ce mot-clé n'est pas pris en charge dans une procédure stockée.

TRANSACTION

Mot-clé facultatif. WORK et TRANSACTION sont synonymes. Aucun des deux n'est pris en charge dans une procédure stockée.

Pour obtenir des informations sur l'utilisation de l'instruction COMMIT dans une procédure stockée, consultez Gestion des transactions.

Exemples

Chacun des exemples suivants valide la transaction en cours sur la base de données :

```
commit;
```

```
commit work;
```

```
commit transaction;
```

COPY

Le chiffrement côté client pour les commandes COPY et UNLOAD ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2025. Si vous avez utilisé le chiffrement côté client avec les commandes COPY et UNLOAD au cours des 12 mois précédant le 30 avril 2025, vous pouvez continuer à utiliser le chiffrement côté client avec les commandes COPY ou UNLOAD jusqu'au 30 avril 2026. Après le 30 avril 2026, vous ne pourrez plus utiliser le chiffrement côté client pour COPY et UNLOAD. Nous vous recommandons de passer au chiffrement côté serveur pour COPY et UNLOAD dès que possible. Si vous utilisez déjà le chiffrement côté serveur pour COPY et UNLOAD, il n'y a aucun changement et vous pouvez continuer à l'utiliser sans modifier vos requêtes. Pour plus d'informations sur le chiffrement pour COPY et UNLOAD, consultez le paramètre ENCRYPTED ci-dessous.

Charge des données dans une table depuis des fichiers de données ou une table Amazon DynamoDB. Les fichiers peuvent être situés dans un compartiment Amazon Simple Storage Service (Amazon S3), un cluster Amazon EMR ou un hôte distant auquel on accède à l'aide d'une connexion SSH (Secure Shell).

> **Note**
>
> Les tables externes d'Amazon Redshift Spectrum sont en lecture seule. Vous ne pouvez pas copier (COPY) une table externe.

La commande COPY ajoute les données d'entrée à la table sous forme de lignes supplémentaires.

La taille maximale d'une seule ligne d'entrée à partir de n'importe quelle source est de 4 Mo.

Rubriques

- Autorisations requises
- Syntaxe de la commande COPY
- Paramètres requis
- Paramètres facultatifs
- Notes d'utilisation et ressources supplémentaires pour la commande COPY
- Exemples de commandes COPY
- TÂCHE DE COPIE
- Description des paramètres de la commande COPY
- Notes d'utilisation
- Exemples de commandes COPY

Autorisations requises

Pour utiliser la commande COPY, vous devez avoir le privilège INSERT pour la table Amazon Redshift.

Syntaxe de la commande COPY

```
COPY table-name
[ column-list ]
FROM data_source
authorization
[ [ FORMAT ] [ AS ] data_format ]
[ parameter [ argument ] [, ... ] ]
```

Vous pouvez effectuer une opération COPY avec aussi peu que trois paramètres : un nom de table, une source de données et l'autorisation d'accéder aux données.

Amazon Redshift étend la fonctionnalité de la commande COPY pour vous permettre de charger les données dans plusieurs formats de données à partir de plusieurs sources de données, de contrôler

l'accès pour charger les données, de gérer les transformations des données et de gérer l'opération de chargement.

Les sections suivantes présentent les paramètres de la commande COPY requis et regroupent les paramètres facultatifs par fonction. Elles décrivent également chaque paramètre et expliquent comment différentes options fonctionnent ensemble. Vous pouvez accéder directement à une description du paramètre à l'aide de la liste alphabétique des paramètres.

Paramètres requis

La commande COPY nécessite trois éléments :

- Table Name
- Data Source
- Authorization

La commande COPY la plus simple utilise le format suivant.

```
COPY table-name
FROM data-source
authorization;
```

L'exemple suivant crée une table nommée CATDEMO et puis charge la table avec des exemples de données à partir d'un fichier de données dans Amazon S3 nommé `category_pipe.txt`.

```
create table catdemo(catid smallint, catgroup varchar(10), catname varchar(10), catdesc
 varchar(50));
```

Dans l'exemple suivant, la source de données de la commande COPY est un fichier de données nommé `category_pipe.txt` dans le dossier `tickit` d'un compartiment Amazon S3 nommé `redshift-downloads`. La commande COPY est autorisée à accéder au compartiment Amazon S3 via un rôle AWS Identity and Access Management (IAM). Si votre cluster comporte un rôle IAM existant avec la permission d'accès à Amazon S3 attaché, vous pouvez remplacer le nom Amazon Resource Name (ARN) de votre rôle dans la commande COPY suivante et l'exécuter.

```
copy catdemo
from 's3://redshift-downloads/tickit/category_pipe.txt'
```

```
iam_role 'arn:aws:iam::<aws-account-id>:role/<role-name>'
region 'us-east-1';
```

Pour obtenir des instructions complètes sur l'utilisation des commandes COPY pour charger des échantillons de données, y compris des instructions pour charger des données provenant d'autres AWS régions, consultez la section Charger des échantillons de données depuis Amazon S3 dans le guide de démarrage Amazon Redshift.

table-name

Nom de la table cible de la commande COPY. La table doit déjà exister dans la base de données. La table peut être temporaire ou permanente. La commande COPY ajoute les nouvelles données d'entrée à toutes les lignes existantes de la table.

FROM data-source

Emplacement des données source à charger dans la table cible. Un fichier manifeste peut être spécifié avec des sources de données.

Le référentiel de données le plus couramment utilisé est un compartiment Amazon S3. Vous pouvez également charger à partir de fichiers de données situés dans un cluster Amazon EMR, une EC2 instance Amazon ou un hôte distant auquel votre cluster peut accéder via une connexion SSH, ou vous pouvez le charger directement à partir d'une table DynamoDB.

- Commande COPY depuis Amazon S3
- Commande COPY depuis Amazon EMR
- Exécution de la commande COPY à partir de l'hôte distant (SSH)
- Commande COPY depuis Amazon DynamoDB

Autorisation

Clause qui indique la méthode utilisée par votre cluster pour l'authentification et l'autorisation d'accéder à d'autres AWS ressources. La commande COPY nécessite une autorisation pour accéder aux données d'une autre AWS ressource, notamment Amazon S3, Amazon EMR, Amazon DynamoDB et Amazon. EC2 Vous pouvez fournir cette autorisation en faisant référence à un rôle IAM qui est attaché à votre cluster ou en fournissant l'ID de clé d'accès et la clé d'accès secrète pour un utilisateur IAM.

- Paramètres d'autorisation
- Contrôle d'accès basé sur les rôles

- Contrôle d’accès basé sur les clés

Paramètres facultatifs

Le cas échéant, vous pouvez spécifier comment la commande COPY mappe les données de champ aux colonnes dans la table cible, définir les attributs de données source pour activer la commande COPY afin de lire et d’analyser correctement les données source et gérer les opérations que la commande COPY effectue pendant le processus de chargement.

- Options de mappage de colonnes
- Paramètres du format de données
- Paramètres de conversion de données
- Opérations de chargement de données

Mappage de colonnes

Par défaut, la commande COPY insère des valeurs de champ dans les colonnes de la table cible dans le même ordre que les champs se présentent dans les fichiers de données. Si l'ordre des colonnes par défaut ne fonctionne pas, vous pouvez spécifier une liste de colonnes ou utiliser des JSONPath expressions pour mapper les champs de données source aux colonnes cibles.

- Column List
- JSONPaths File

Paramètres du format de données

Vous pouvez charger les données à partir de fichiers texte au format JSON, dans des fichiers de valeurs séparées par des virgules, par des caractères, à largeur fixe (CSV), ou à partir de fichiers Avro.

Par défaut, la commande COPY attend que les données source se trouvent dans des fichiers texte UTF-8 séparés par des caractères. Le délimiteur par défaut est une barre verticale (|). Si les données source sont dans un autre format, utilisez les paramètres suivants pour spécifier le format des données.

- FORMAT
- CSV

- DELIMITER
- FIXEDWIDTH
- SHAPEFILE
- AVRO
- JSON
- ENCRYPTED
- BZIP2
- GZIP
- LZOP
- PARQUET
- ORC
- ZSTD

Paramètres de conversion de données

Lorsqu'elle charge la table, la commande COPY tente implicitement de convertir les chaînes dans les données source vers le type de données de la colonne cible. Si vous devez spécifier une conversion qui est différente du comportement par défaut, ou si la conversion par défaut entraîne des erreurs, vous pouvez gérer les conversions de données en spécifiant les paramètres suivants.

- ACCEPTANYDATE
- ACCEPTINVCHARS
- BLANKSASNULL
- DATEFORMAT
- EMPTYASNULL
- ENCODING
- ESCAPE
- EXPLICIT_IDS
- FILLRECORD
- IGNOREBLANKLINES
- IGNOREHEADER

- NULL AS
- REMOVEQUOTES
- ROUNDEC
- TIMEFORMAT
- TRIMBLANKS
- TRUNCATECOLUMNS

Opérations de chargement de données

Gérez le comportement par défaut de l'opération de chargement pour le dépannage ou pour réduire les temps de chargement en spécifiant les paramètres suivants.

- COMPROWS
- COMPUPDATE
- IGNOREALLERRORS
- MAXERROR
- NOLOAD
- STATUPDATE

Notes d'utilisation et ressources supplémentaires pour la commande COPY

Pour plus d'informations sur la façon d'utiliser la commande COPY, consultez les rubriques suivantes :

- Notes d'utilisation
- Didacticiel : chargement des données à partir d'Amazon S3
- Bonnes pratiques de chargement des données sur Amazon Redshift
- Chargement de tables à l'aide de la commande COPY
  - Chargement des données à partir d'Amazon S3
  - Chargement de données à partir d'Amazon EMR
  - Chargement des données à partir des hôtes distants
  - Chargement de données à partir d'une table Amazon DynamoDB
- Résolution des problèmes de chargement de données

Exemples de commandes COPY

Pour d'autres exemples montrant comment utiliser COPY à partir de différentes sources, dans des formats disparates et avec différentes options de copie, consultez Exemples de commandes COPY.

TÂCHE DE COPIE

Pour plus d'informations sur l'utilisation de cette commande, consultezCréez une intégration d'événements S3 pour copier automatiquement des fichiers depuis des compartiments Amazon S3.

Gère les commandes COPY qui chargent les données dans une table. La commande COPY JOB est une extension de la commande COPY et automatise le chargement des données à partir des compartiments Amazon S3. Lorsque vous créez une tâche COPY, Amazon Redshift détecte quand de nouveaux fichiers Amazon S3 sont créés dans un chemin spécifié, puis les charge automatiquement sans votre intervention. Les mêmes paramètres que ceux utilisés dans la commande COPY d'origine sont utilisés lors du chargement des données. Amazon Redshift assure le suivi des fichiers chargés (en fonction du nom de fichier) pour vérifier qu'ils ne sont chargés qu'une seule fois.

> **Note**
>
> Pour plus d'informations sur la commande COPY, notamment sur l'utilisation, les paramètres et les autorisations, consultez COPY.

Autorisation obligatoire

Pour exécuter la commande COPY d'une tâche COPY JOB, vous devez disposer du privilège INSERT sur la table en cours de chargement.

Le rôle IAM spécifié avec la commande COPY doit être autorisé à accéder aux données à charger. Pour de plus amples informations, veuillez consulter Autorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY.

Syntaxe

Créez une tâche de copie. Les paramètres de la commande COPY sont enregistrés avec la tâche de copie.

Vous ne pouvez pas exécuter COPY JOB CREATE dans le cadre d'un bloc de transactions.

```
COPY copy-command JOB CREATE job-name
[AUTO ON | OFF]
```

Modifiez la configuration d'une tâche de copie.

```
COPY JOB ALTER job-name
[AUTO ON | OFF]
```

Exécutez une tâche de copie. Les paramètres de la commande COPY stockés sont utilisés.

```
COPY JOB RUN job-name
```

Répertoriez toutes les tâches de copie.

```
COPY JOB LIST
```

Afficher les détails d'une tâche de copie.

```
COPY JOB SHOW job-name
```

Supprimez une tâche de copie.

Vous ne pouvez pas exécuter COPY JOB DROP dans le cadre d'un bloc de transactions.

```
COPY JOB DROP job-name
```

Paramètres

copy-command

Une commande COPY qui charge les données d'Amazon S3 vers Amazon Redshift. La clause contient des paramètres COPY qui définissent le compartiment Amazon S3, la table cible, le rôle IAM et d'autres paramètres utilisés lors du chargement des données. Tous les paramètres de commande COPY pour un chargement de données Amazon S3 sont pris en charge, à l'exception des suivants :

- La tâche COPY JOB n'ingère pas les fichiers préexistants dans le dossier pointé par la commande COPY. Seuls les fichiers créés après l'horodatage de création de COPY JOB sont ingérés.

- Vous ne pouvez pas spécifier de commande COPY avec les options MAXERROR ou IGNOREALLERRORS.
- Vous ne pouvez pas spécifier un fichier manifeste. COPY JOB nécessite un emplacement Amazon S3 désigné pour surveiller les fichiers nouvellement créés.
- Vous ne pouvez pas spécifier de commande COPY avec des types d'autorisation tels que des clés d'accès et des clés secrètes. Seules les commandes COPY qui utilisent le paramètre `IAM_ROLE` pour l'autorisation sont prises en charge. Pour de plus amples informations, veuillez consulter Paramètres d'autorisation.
- COPY JOB ne prend pas en charge le rôle IAM par défaut associé au cluster. Vous devez spécifier le `IAM_ROLE` dans la commande COPY.

Pour de plus amples informations, veuillez consulter Commande COPY depuis Amazon S3.

job-name

Nom de la tâche utilisé pour référencer la tâche COPY. Le nom de la tâche ne peut pas contenir de tiret (-).

[AUTO ON | OFF]

Clause indiquant si les données Amazon S3 sont automatiquement chargées dans les tables Amazon Redshift.

- Sur `ON`, Amazon Redshift surveille le chemin Amazon S3 source pour les fichiers nouvellement créés et, s'il en trouve, une commande COPY est exécutée avec les paramètres COPY dans la définition de la tâche. Il s'agit de l'option par défaut.
- Sur `OFF`, Amazon Redshift n'exécute pas automatiquement COPY JOB.

Notes d'utilisation

Les options de la commande COPY ne sont validées qu'au moment de l'exécution. Par exemple, un `IAM_ROLE` ou une source de données Amazon S3 non valide entraîne des erreurs d'exécution lorsque COPY JOB démarre.

Si le cluster est suspendu, COPY JOBS ne sont pas exécutées.

Pour interroger les fichiers de commandes COPY chargés et les erreurs de chargement, consultez STL_LOAD_COMMITS, STL_LOAD_ERRORS et STL_LOADERROR_DETAIL. Pour de plus amples informations, veuillez consulter Vérification que les données ont été chargées correctement.

Exemples

L'exemple suivant montre comment créer COPY JOB pour charger les données d'un compartiment Amazon S3.

```
COPY public.target_table
FROM 's3://amzn-s3-demo-bucket/staging-folder'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyLoadRoleName'
JOB CREATE my_copy_job_name
AUTO ON;
```

Description des paramètres de la commande COPY

COPY possède de nombreux paramètres qui peuvent être utilisés dans de nombreuses situations. Toutefois, tous les paramètres ne sont pas pris en charge dans chaque situation. Par exemple, pour le chargement à partir de fichiers ORC ou PARQUET, le nombre de paramètres pris en charge est limité. Pour de plus amples informations, veuillez consulter COPY depuis les formats de données en colonnes.

Rubriques

- Sources de données
- Paramètres d'autorisation
- Options de mappage de colonnes
- Paramètres du format de données
- Paramètres de compression de fichier
- Paramètres de conversion de données
- Opérations de chargement de données
- Liste alphabétique des paramètres

Sources de données

Vous pouvez charger des données à partir de fichiers texte dans un compartiment Amazon S3, dans un cluster Amazon EMR, ou sur un hôte distant auquel votre cluster peut accéder à l'aide d'une connexion SSH. Vous pouvez également charger les données directement à partir d'une table DynamoDB.

La taille maximale d'une seule ligne d'entrée à partir de n'importe quelle source est de 4 Mo.

Pour exporter les données d'une table dans un ensemble de fichiers dans une instance Amazon S3, utilisez la commande UNLOAD.

Rubriques

- Commande COPY depuis Amazon S3
- Commande COPY depuis Amazon EMR
- Exécution de la commande COPY à partir de l'hôte distant (SSH)
- Commande COPY depuis Amazon DynamoDB

Commande COPY depuis Amazon S3

Pour charger des données à partir de fichiers situés dans un ou plusieurs compartiments S3, utilisez la clause FROM pour indiquer comment la commande COPY localise les fichiers dans Amazon S3. Vous pouvez fournir le chemin d'objet aux fichiers de données dans le cadre de la clause FROM, ou vous pouvez fournir l'emplacement d'un fichier manifeste qui contient une liste de chemins d'objets Amazon S3. L'exécution de la commande COPY à partir d'Amazon S3 utilise une connexion HTTPS. Assurez-vous que les plages d'adresses IP S3 sont ajoutées à votre liste des autorisations. Pour plus d'informations sur les plages d'adresses IP S3 requises, consultez Isolement de réseau.

> **Important**
>
> Si les compartiments Amazon S3 qui contiennent les fichiers de données ne résident pas dans la même AWS région que votre cluster, vous devez utiliser le REGION paramètre pour spécifier la région dans laquelle se trouvent les données.

Rubriques

- Syntaxe
- Exemples
- Paramètres facultatifs
- Paramètres non pris en charge

Syntaxe

```
FROM { 's3://objectpath' | 's3://manifest_file' }
authorization
```

```
| MANIFEST
| ENCRYPTED
| REGION [AS] 'aws-region'
| optional-parameters
```

Exemples

L'exemple suivant utilise un chemin d'objet pour charger les données à partir d'Amazon S3.

```
copy customer
from 's3://amzn-s3-demo-bucket/customer'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

L'exemple suivant utilise un fichier manifeste pour charger les données à partir d'Amazon S3.

```
copy customer
from 's3://amzn-s3-demo-bucket/cust.manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest;
```

Paramètres

FROM

Source des données à charger. Pour plus d'informations sur l'encodage du fichier Amazon S3, consultez Paramètres de conversion de données.

's3://copy_from_s3_objectpath'

Spécifie le chemin d'accès aux objets Amazon S3 contenant les données (par exemple `'s3://amzn-s3-demo-bucket/custdata.txt'`). Le paramètre s3://copy_from_s3_objectpath peut faire référence à un seul fichier ou à un ensemble d'objets ou de dossiers ayant le même préfixe de clé. Par exemple, le nom `custdata.txt` est un préfixe de clé qui fait référence à un certain nombre de fichiers physiques : `custdata.txt`, `custdata.txt.1`, `custdata.txt.2`, `custdata.txt.bak` et ainsi de suite. Le préfixe de clé peut également faire référence à un certain nombre de dossiers. Par exemple, `'s3://amzn-s3-demo-bucket/custfolder'` fait référence aux dossiers `custfolder`, `custfolder_1`, `custfolder_2` et ainsi de suite. Si un préfixe de clé fait référence à plusieurs dossiers, tous les fichiers dans les dossiers sont chargés. Si un préfixe de clé correspond à un fichier et à un dossier, par exemple `custfolder.log`, COPY tente de charger le fichier également. Si un préfixe de clé risque d'entraîner le chargement

de fichiers indésirables par COPY, utilisez un fichier manifeste. Pour plus d'informations, consultez copy_from_s3_manifest_file, ci-après.

> **Important**
>
> Si le compartiment S3 qui contient les fichiers de données ne réside pas dans la même AWS région que votre cluster, vous devez utiliser le REGION paramètre pour spécifier la région dans laquelle se trouvent les données.

Pour de plus amples informations, veuillez consulter Chargement des données à partir d'Amazon S3.

's3://copy_from_s3_manifest_file'

Spécifie la clé de l'objet Amazon S3 d'un fichier manifeste qui répertorie les fichiers de données à charger. L'argument 's3://copy_from_s3_manifest_file' doit explicitement faire référence à un seul fichier, par exemple, `'s3://amzn-s3-demo-bucket/manifest.txt'`. Il ne peut pas faire référence à un préfixe de clé.

Le manifeste est un fichier texte au format JSON qui répertorie l'URL de chaque fichier qui doit être chargé à partir d'Amazon S3. L'URL inclut le nom du compartiment et le chemin d'objet complet du fichier. Les fichiers spécifiés dans le manifeste peuvent se trouver dans différents compartiments, mais tous les compartiments doivent se trouver dans la même AWS région que le cluster Amazon Redshift. Si un fichier est répertorié deux fois, le fichier est chargé deux fois. L'exemple suivant illustre le format JSON pour un manifeste qui charge trois fichiers.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket1/custdata.1","mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket1/custdata.2","mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket2/custdata.1","mandatory":false}
  ]
}
```

Les guillemets doubles sont nécessaires et doivent être des guillemets simples (0x22), ni culbutés, ni courbes. Chaque entrée dans le manifeste peut éventuellement inclure un indicateur `mandatory`. Si `mandatory` est défini sur `true`, la commande COPY s'arrête si elle ne trouve pas le fichier pour cette entrée ; dans le cas contraire, la commande COPY se poursuit. La valeur par défaut de `mandatory` est `false`.

Lors du chargement des fichiers de données au format ORC or Parquet, le champ `meta` est obligatoire, comme illustré dans l'exemple suivant.

```
{
   "entries":[
      {
         "url":"s3://amzn-s3-demo-bucket1/orc/2013-10-04-custdata",
         "mandatory":true,
         "meta":{
            "content_length":99
         }
      },
      {
         "url":"s3://amzn-s3-demo-bucket2/orc/2013-10-05-custdata",
         "mandatory":true,
         "meta":{
            "content_length":99
         }
      }
   ]
}
```

Le fichier manifeste ne doit pas être chiffré ou compressé, même si les options ENCRYPTED, GZIP, LZOP ou ZSTD sont spécifiées. BZIP2 La commande COPY renvoie une erreur si le fichier manifeste spécifié est introuvable ou si le fichier manifeste n'est pas correctement formé.

Si un fichier manifeste est utilisé, le paramètre MANIFEST doit être spécifié avec la commande COPY. Si le paramètre MANIFEST n'est pas spécifié, la commande COPY présume que le fichier spécifié avec FROM est un fichier de données.

Pour plus d'informations, consultez Chargement des données à partir d'Amazon S3.

authorization

La commande COPY nécessite une autorisation pour accéder aux données d'une autre AWS ressource, notamment Amazon S3, Amazon EMR, Amazon DynamoDB et Amazon. EC2 Vous pouvez fournir cette autorisation en référençant un rôle AWS Identity and Access Management (IAM) attaché à votre cluster (contrôle d'accès basé sur les rôles) ou en fournissant les informations d'identification d'accès d'un utilisateur (contrôle d'accès basé sur des clés). Pour plus de sécurité et de flexibilité, nous recommandons d'utiliser contrôle d'accès basé sur les rôles IAM. Pour plus d'informations, consultez Paramètres d'autorisation.

MANIFEST

Spécifie qu'un manifeste est utilisé pour identifier les fichiers de données à charger à partir d'Amazon S3. Si le paramètre MANIFEST est utilisé, la commande COPY charge des données depuis les fichiers répertoriés dans le manifeste référencé par 's3://copy_from_s3_manifest_file'. Si le fichier manifeste n'est pas trouvé ou est dans un format incorrect, la commande COPY échoue. Pour plus d'informations, consultez Utilisation d'un manifeste pour spécifier les fichiers de données.

ENCRYPTED

Une clause qui spécifie que les fichiers d'entrée sur Amazon S3 sont chiffrés à l'aide du chiffrement côté client avec des clés gérées par le client. Pour plus d'informations, consultez Chargement de fichiers de données chiffrés à partir d'Amazon S3. N'indiquez pas ENCRYPTED si les fichiers d'entrée sont chiffrés à l'aide du chiffrement côté serveur Amazon S3 (SSE-KMS ou SSE-S3). La commande COPY lit automatiquement les fichiers chiffrés côté serveur.

Si vous spécifiez le paramètre ENCRYPTED, vous devez également spécifier le paramètre MASTER_SYMMETRIC_KEY ou inclure la valeur **`master_symmetric_key`** dans la chaîne the section called "Utilisation des informations d'identification".

Si les fichiers chiffrés sont au format compressé, ajoutez le paramètre GZIP, LZOP ou BZIP2 ZSTD.

Les fichiers manifestes et JSONPaths les fichiers ne doivent pas être chiffrés, même si l'option ENCRYPTED est spécifiée.

MASTER_SYMMETRIC_KEY 'root_key'

La clé symétrique racine qui a été utilisée pour chiffrer les fichiers de données sur Amazon S3. Si la clé MASTER_SYMMETRIC_KEY est spécifiée, le paramètre ENCRYPTED doit également être spécifié. La clé MASTER_SYMMETRIC_KEY ne peut pas être utilisée avec le paramètre CREDENTIALS. Pour de plus amples informations, veuillez consulter Chargement de fichiers de données chiffrés à partir d'Amazon S3.

Si les fichiers chiffrés sont au format compressé, ajoutez le paramètre GZIP, LZOP ou BZIP2 ZSTD.

REGION [AS] 'aws-region'

Spécifie la AWS région dans laquelle se trouvent les données sources. REGION est nécessaire pour exécuter la commande COPY depuis un compartiment Amazon S3 ou une table DynamoDB

lorsque la ressource AWS qui contient les données ne se trouve pas dans la même région que le cluster Amazon Redshift.

La valeur pour aws_region doit correspondre à une région répertoriée dans le tableau Régions et points de terminaison Amazon Redshift.

Si le paramètre REGION est spécifié, toutes les ressources, y compris un fichier manifeste ou plusieurs compartiments Amazon S3, doivent se trouver dans la région spécifiée.

> **Note**
>
> Le transfert de données entre les régions engage des frais supplémentaires pour le compartiment Amazon S3 ou la table DynamoDB qui contient les données. Pour plus d'informations sur la tarification, consultez les sections Transfert de données sortantes d'Amazon S3 vers une autre AWS région sur la page de tarification d'Amazon S3 et Transfert de données sortantes sur la page de tarification d'Amazon DynamoDB.

Par défaut, la commande COPY part du principe que les données se trouvent dans la même région que le cluster Amazon Redshift.

Paramètres facultatifs

Vous pouvez éventuellement spécifier les paramètres suivants avec la commande COPY à partir d'Amazon S3 :

- Options de mappage de colonnes
- Paramètres du format de données
- Paramètres de conversion de données
- Opérations de chargement de données

Paramètres non pris en charge

Vous ne pouvez pas utiliser les paramètres suivants avec la commande COPY à partir d'Amazon S3 :

- SSH
- READRATIO

Commande COPY depuis Amazon EMR

Vous pouvez utiliser la commande COPY pour charger des données en parallèle à partir d'un cluster Amazon EMR configuré pour écrire des fichiers texte dans le système de fichiers distribué Hadoop (HDFS) du cluster sous la forme de fichiers à largeur fixe, de fichiers séparés par des caractères, de fichiers CSV, de fichiers au format JSON ou de fichiers Avro.

Rubriques

- Syntaxe
- exemple
- Paramètres
- Paramètres pris en charge
- Paramètres non pris en charge

Syntaxe

```
FROM 'emr://emr_cluster_id/hdfs_filepath'
authorization
[ optional_parameters ]
```

exemple

L'exemple suivant charge des données depuis un cluster Amazon EMR.

```
copy sales
from 'emr://j-SAMPLE2B500FC/myoutput/part-*'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Paramètres

FROM

Source des données à charger.

'emr://emr_cluster_id/hdfs_file_path'

Identifiant unique du cluster Amazon EMR et chemin d'accès au fichier HDFS faisant référence aux fichiers de données pour la commande COPY. Les noms de fichiers de données HDFS ne doivent pas comporter les caractères génériques suivants : l'astérisque (*) et le point d'interrogation (?).

> **Note**
>
> Le cluster Amazon EMR doit continuer de s'exécuter jusqu'à la fin de l'opération COPY. Si l'un des fichiers de données HDFS est modifié ou supprimé avant la fin de l'opération COPY, vous pouvez avoir des résultats inattendus ou l'opération COPY peut échouer.

Vous pouvez utiliser les caractères génériques astérisque (*) et point d'interrogation (?) dans le cadre de l'argument hdfs_file_path pour spécifier le chargement de plusieurs fichiers. Par exemple, `'emr://j-SAMPLE2B500FC/myoutput/part*'` identifie les fichiers `part-0000`, `part-0001`, et ainsi de suite. Si le chemin d'accès ne contient pas de caractères génériques, il est traité comme un littéral de chaîne. Si vous spécifiez uniquement un nom de dossier, COPY tente de charger tous les fichiers dans le dossier.

> **Important**
>
> Si vous utilisez des caractères génériques ou uniquement le nom du dossier, vérifiez qu'aucun fichier indésirable ne sera chargé. Par exemple, certains processus peuvent écrire un fichier journal sur le dossier de sortie.

Pour plus d'informations, consultez Chargement de données à partir d'Amazon EMR.

authorization

La commande COPY nécessite une autorisation pour accéder aux données d'une autre AWS ressource, notamment Amazon S3, Amazon EMR, Amazon DynamoDB et Amazon. EC2 Vous pouvez fournir cette autorisation en référençant un rôle AWS Identity and Access Management (IAM) attaché à votre cluster (contrôle d'accès basé sur les rôles) ou en fournissant les informations d'identification d'accès d'un utilisateur (contrôle d'accès basé sur des clés). Pour plus de sécurité et de flexibilité, nous recommandons d'utiliser contrôle d'accès basé sur les rôles IAM. Pour plus d'informations, consultez Paramètres d'autorisation.

Paramètres pris en charge

Vous pouvez éventuellement spécifier les paramètres suivants avec la commande COPY à partir d'Amazon EMR :

- Options de mappage de colonnes

- Paramètres du format de données
- Paramètres de conversion de données
- Opérations de chargement de données

Paramètres non pris en charge

Vous ne pouvez pas utiliser les paramètres suivants avec la commande COPY à partir d'Amazon EMR :

- ENCRYPTED
- MANIFEST
- REGION
- READRATIO
- SSH

Exécution de la commande COPY à partir de l'hôte distant (SSH)

Vous pouvez utiliser la commande COPY pour charger des données en parallèle depuis un ou plusieurs hôtes distants, tels que des instances Amazon Elastic Compute Cloud (Amazon EC2) ou d'autres ordinateurs. La commande COPY se connecte aux hôtes distants à l'aide de SSH (Secure Shell) et exécute les commandes sur les hôtes distants pour générer la sortie de texte. L'hôte distant peut être une instance EC2 Linux ou un autre ordinateur Unix ou Linux configuré pour accepter les connexions SSH. Amazon Redshift peut se connecter à plusieurs hôtes et ouvrir plusieurs connexions SSH à chaque hôte. Amazon Redshift envoie une commande unique lors de chaque connexion pour générer la sortie de texte sur la sortie standard de l'hôte, qu'Amazon Redshift lit alors comme un fichier texte.

Utilisez la clause FROM pour spécifier la clé d'objet Amazon S3 pour le fichier manifeste qui fournit les informations que la commande COPY utilise pour ouvrir des connexions SSH et exécuter les commandes à distance.

Rubriques

- Syntaxe
- Exemples
- Paramètres
- Paramètres facultatifs

- Paramètres non pris en charge

> **Important**
>
> Si le compartiment S3 contenant le fichier manifeste ne réside pas dans la même région AWS que votre cluster, vous devez utiliser le paramètre REGION pour spécifier la région dans laquelle le compartiment se trouve.

Syntaxe

```
FROM 's3://'ssh_manifest_file' }
authorization
SSH
| optional-parameters
```

Exemples

L'exemple suivant utilise un fichier manifeste pour charger les données à partir d'un hôte distant à l'aide de SSH.

```
copy sales
from 's3://amzn-s3-demo-bucket/ssh_manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
ssh;
```

Paramètres

FROM

Source des données à charger.

's3://copy_from_ssh_manifest_file'

La commande COPY peut se connecter à plusieurs hôtes à l'aide de SSH et créer plusieurs connexions SSH à chaque hôte. COPY exécute une commande via chaque connexion hôte, puis charge la sortie à partir des commandes en parallèle de la table. L'argument s3://copy_from_ssh_manifest_file spécifie la clé d'objet Amazon S3 du fichier manifeste qui fournit les informations que la commande COPY utilise pour ouvrir les connexions SSH et exécuter les commandes à distance.

L'argument s3://copy_from_ssh_manifest_file doit explicitement faire référence à un seul fichier ; il ne peut pas être un préfixe de clé. Voici un exemple:

```
's3://amzn-s3-demo-bucket/ssh_manifest.txt'
```

Le fichier manifeste est un fichier texte au format JSON qu'Amazon Redshift utilise pour se connecter à l'hôte. Le fichier manifeste spécifie les points de terminaison hôte SSH et les commandes qui seront exécutées sur les hôtes pour renvoyer les données à Amazon Redshift. Le cas échéant, vous pouvez inclure la clé publique de l'hôte, le nom d'utilisateur de connexion et un indicateur obligatoire pour chaque entrée. L'exemple suivant montre un fichier manifeste qui crée deux connexions SSH :

```
{
    "entries": [
     {"endpoint":"<ssh_endpoint_or_IP>",
           "command": "<remote_command>",
           "mandatory":true,
           "publickey": "<public_key>",
           "username": "<host_user_name>"},
     {"endpoint":"<ssh_endpoint_or_IP>",
           "command": "<remote_command>",
           "mandatory":true,
           "publickey": "<public_key>",
           "username": "<host_user_name>"}
     ]
}
```

Le fichier manifeste contient une construction `"entries"` pour chaque connexion SSH. Vous pouvez avoir plusieurs connexions à un seul hôte ou plusieurs connexions à plusieurs hôtes. Les guillemets doubles sont obligatoires comme illustré, aussi bien pour les noms de champ que pour les valeurs. Les guillemets doivent être des guillemets simples (0x22), ni culbutés, ni courbes. La seule valeur qui n'a pas besoin de guillemets doubles est la valeur booléenne `true` ou `false` pour le champ `"mandatory"`.

La liste suivante décrit les champs dans le fichier manifeste.

point de terminaison

L'adresse URL ou l'adresse IP de l'hôte, par exemple, `"ec2-111-222-333.compute-1.amazonaws.com"` ou `"198.51.100.0"`.

command

La commande doit être exécutée par l'hôte pour générer la sortie de texte ou la sortie binaire au format gzip, lzop, bzip2 ou zstd. La commande peut être n'importe quelle commande que l'utilisateur « host_user_name » est autorisé à exécuter. La commande peut être aussi simple que l'impression d'un fichier, ou elle bien peut interroger une base de données ou lancer un script. Les sorties (fichier texte, fichier binaire gzip, fichier binaire lzop ou fichier binaire bzip2) doivent être sous une forme que la commande COPY Amazon Redshift peut intégrer. Pour plus d'informations, consultez Préparation de vos données d'entrée.

publickey

(Facultatif) La clé publique de l'hôte. Si la clé est fournie, Amazon Redshift l'utilise pour identifier l'hôte. Si la clé publique n'est pas fournie, Amazon Redshift n'essaie pas d'identifier l'hôte. Par exemple, si la clé publique l'hôte distant est `ssh-rsa AbcCbaxxx…Example root@amazon.com`, tapez le texte suivant dans le champ de clé publique : `"AbcCbaxxx…Example"`

mandatory

(Facultatif) Clause qui indique si la commande COPY doit échouer en cas d'échec de la tentative de connexion. La valeur par défaut est `false`. Si Amazon Redshift n'établit pas au moins une connexion, la commande COPY échoue.

nom d'utilisateur

(Facultatif) Nom d'utilisateur qui sera utilisé pour vous connecter au système hôte et exécuter la commande à distance. Le nom de connexion d'utilisateur doit être le même que la connexion qui a été utilisée pour ajouter la clé publique du cluster Amazon Redshift au fichier de clés autorisé de l'hôte. Le nom d'utilisateur par défaut est `redshift`.

Pour plus d'informations sur la création d'un fichier manifeste, consultez Processus de chargement de données.

Pour exécuter la commande COPY d'un hôte distant, le paramètre SSH doit être spécifié avec la commande COPY. Si le paramètre SSH n'est pas spécifié, la commande COPY suppose que le fichier spécifié avec FROM est un fichier de données et échoue.

Si vous utilisez la compression automatique, la commande COPY effectue deux opérations de lecture des données, ce qui signifie qu'elle va exécuter la commande à distance à deux reprises. La première opération de lecture vise à fournir un échantillon de données pour l'analyse de la

compression, et la deuxième opération de lecture charge réellement les données. Si l'exécution de la commande à distance à deux reprises est susceptible d'entraîner un problème, vous devez désactiver la compression automatique. Pour désactiver la compression automatique, exécutez la commande COPY avec le paramètre COMPUPDATE défini sur OFF. Pour plus d'informations, consultez Chargement des tables avec compression automatique.

Pour connaître les procédures détaillées de l'utilisation de la commande COPY dans SSH, consultez Chargement des données à partir des hôtes distants.

authorization

La commande COPY nécessite une autorisation pour accéder aux données d'une autre AWS ressource, notamment Amazon S3, Amazon EMR, Amazon DynamoDB et Amazon. EC2 Vous pouvez fournir cette autorisation en référençant un rôle AWS Identity and Access Management (IAM) attaché à votre cluster (contrôle d'accès basé sur les rôles) ou en fournissant les informations d'identification d'accès d'un utilisateur (contrôle d'accès basé sur des clés). Pour plus de sécurité et de flexibilité, nous recommandons d'utiliser contrôle d'accès basé sur les rôles IAM. Pour plus d'informations, consultez Paramètres d'autorisation.

SSH

Clause qui spécifie que les données doivent être chargées à partir d'un hôte distant à l'aide du protocole SSH. Si vous spécifiez SSH, vous devez également fournir un fichier manifeste à l'aide de l'argument s3://copy_from_ssh_manifest_file.

> **Note**
>
> Si vous utilisez SSH pour effectuer une copie à partir d'un hôte à l'aide d'une adresse IP privée dans un VPC distant, le routage VPC amélioré doit être activé pour le VPC. Pour plus d'informations sur le routage VPC amélioré, consultez Routage VPC amélioré Amazon Redshift.

Paramètres facultatifs

Vous pouvez éventuellement spécifier les paramètres suivants avec la commande COPY à partir de SSH :

- Options de mappage de colonnes
- Paramètres du format de données

- Paramètres de conversion de données
- Opérations de chargement de données

Paramètres non pris en charge

Vous ne pouvez pas utiliser les paramètres suivants avec la commande COPY à partir de SSH :

- ENCRYPTED
- MANIFEST
- READRATIO

Commande COPY depuis Amazon DynamoDB

Pour charger les données à partir d'une table DynamoDB existante, utilisez la clause FROM pour spécifier le nom de la table DynamoDB.

Rubriques

- Syntaxe
- Exemples
- Paramètres facultatifs
- Paramètres non pris en charge

> **Important**
>
> Si la table DynamoDB ne réside pas dans la même région que votre cluster Amazon Redshift, vous devez utiliser le paramètre REGION pour spécifier la région dans laquelle les données se trouvent.

Syntaxe

```
FROM 'dynamodb://table-name'
authorization
READRATIO ratio
| REGION [AS] 'aws_region'
| optional-parameters
```

Exemples

L’exemple suivant charge des données à partir d’une table DynamoDB.

```
copy favoritemovies from 'dynamodb://ProductCatalog'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
readratio 50;
```

Paramètres

FROM

Source des données à charger.

’dynamodb://table-name’

Nom de la table DynamoDB qui contient les données, par exemple, `'dynamodb://ProductCatalog'`. Pour plus d’informations sur la manière dont les attributs DynamoDB sont mappés aux colonnes Amazon Redshift, consultez Chargement de données à partir d’une table Amazon DynamoDB.

Le nom d'une table DynamoDB est propre à AWS un compte, qui est identifié par AWS les informations d'identification d'accès.

authorization

La commande COPY nécessite une autorisation pour accéder aux données d'une autre AWS ressource, notamment Amazon S3, Amazon EMR, DynamoDB et Amazon. EC2 Vous pouvez fournir cette autorisation en référençant un rôle AWS Identity and Access Management (IAM) attaché à votre cluster (contrôle d'accès basé sur les rôles) ou en fournissant les informations d'identification d'accès d'un utilisateur (contrôle d'accès basé sur des clés). Pour plus de sécurité et de flexibilité, nous recommandons d’utiliser contrôle d’accès basé sur les rôles IAM. Pour plus d'informations, consultez Paramètres d’autorisation.

READRATIO [AS] ratio

Pourcentage du débit alloué à la table DynamoDB à utiliser pour le chargement des données. READRATIO est nécessaire pour exécuter la commande COPY à partir de DynamoDB. Il ne peut pas être utilisé pour exécuter la commande COPY à partir d’Amazon S3. Nous vous recommandons vivement de définir le ratio avec une valeur inférieure au débit moyen alloué non utilisé. Les valeurs valides sont des nombres entiers compris entre 1 et 200.

> **Important**
>
> La définition de READRATIO sur 100 ou plus active Amazon Redshift pour tirer parti de l'intégralité du débit alloué de la table DynamoDB, ce qui dégrade considérablement les performances des opérations de lecture simultanées par rapport à la même table pendant la séance COPY. Le trafic d'écriture n'est pas affecté. Les valeurs supérieures à 100 sont autorisées pour résoudre les scénarios rares où Amazon Redshift ne respecte pas le débit alloué de la table. Si vous chargez les données de DynamoDB vers Amazon Redshift en permanence, pensez à organiser vos tables DynamoDB sous forme de série chronologique pour séparer le trafic en direct de l'opération COPY.

Paramètres facultatifs

Vous pouvez éventuellement spécifier les paramètres suivants avec la commande COPY à partir d'Amazon DynamoDB :

- Options de mappage de colonnes
- Les paramètres de conversion de données suivants sont pris en charge :
  - ACCEPTANYDATE
  - BLANKSASNULL
  - DATEFORMAT
  - EMPTYASNULL
  - ROUNDEC
  - TIMEFORMAT
  - TRIMBLANKS
  - TRUNCATECOLUMNS
- Opérations de chargement de données

Paramètres non pris en charge

Vous ne pouvez pas utiliser les paramètres suivants avec la commande COPY à partir de DynamoDB :

- Paramètres du format de toutes les données
- ESCAPE

- FILLRECORD
- IGNOREBLANKLINES
- IGNOREHEADER
- NULL
- REMOVEQUOTES
- ACCEPTINVCHARS
- MANIFEST
- ENCRYPTED

Paramètres d’autorisation

La commande COPY nécessite une autorisation pour accéder aux données d'une autre AWS ressource, notamment Amazon S3, Amazon EMR, Amazon DynamoDB et Amazon. EC2 Vous fournissez cette autorisation en référençant un rôle AWS Identity and Access Management (IAM) qui est attaché à votre cluster (contrôle d’accès basé sur le rôle). Vous pouvez chiffrer vos données de chargement sur Amazon S3.

Les rubriques suivantes fournissent plus de détails et des exemples d’options d’authentification:

- Autorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY
- Contrôle d’accès basé sur les rôles
- Contrôle d’accès basé sur les clés

Utilisez l’un des paramètres suivants pour autoriser la commande COPY :

- the section called “Utilisation de IAM_ROLE” paramètre
- the section called “Utilisation de ACCESS_KEY_ID et SECRET_ACCESS_KEY” paramètres
- the section called “Utilisation des informations d'identification”Clause

Utilisation du paramètre IAM_ROLE

IAM_ROLE

Utilisez le mot clé par défaut pour qu’Amazon Redshift utilise le rôle IAM défini par défaut et associé au cluster lorsque la commande COPY s’exécute.

Utilisez l'Amazon Resource Name (ARN) d'un rôle IAM que votre cluster utilise pour l'authentification et l'autorisation. Si vous spécifiez IAM_ROLE, vous ne pouvez pas utiliser ACCESS_KEY_ID et SECRET_ACCESS_KEY, SESSION_TOKEN ni CREDENTIALS.

L'exemple suivant montre la syntaxe du paramètre IAM_ROLE.

```
IAM_ROLE { default | 'arn:aws:iam::<Compte AWS-id>:role/<role-name>' }
```

Pour de plus amples informations, veuillez consulter Contrôle d'accès basé sur les rôles.

Utilisation des paramètres ACCESS_KEY_ID et SECRET_ACCESS_KEY

ID DE CLÉ D'ACCÈS, CLÉ D'ACCÈS SECRÈTE

Cette méthode d'autorisation n'est pas recommandée.

> **Note**
>
> Plutôt que de fournir des informations d'identification d'accès sous forme de texte brut, nous vous recommandons vivement d'utiliser l'authentification basée sur les rôles en spécifiant le paramètre IAM_ROLE. Pour de plus amples informations, veuillez consulter Contrôle d'accès basé sur les rôles.

SESSION_TOKEN

Le jeton de séance à utiliser avec les informations d'identification d'accès temporaires. Lorsque SESSION_TOKEN est spécifié, vous devez également utiliser ACCESS_KEY_ID et SECRET_ACCESS_KEY afin de fournir des informations d'identification de clé d'accès temporaire. Si vous spécifiez SESSION_TOKEN, vous ne pouvez utiliser ni IAM_ROLE, ni CREDENTIALS. Pour plus d'informations, consultez informations d'identification de sécurité temporaires dans le Guide de l'utilisateur IAM.

> **Note**
>
> Plutôt que de créer des informations d'identification de sécurité temporaires, nous vous recommandons vivement d'utiliser l'authentification basée sur les rôles. Lorsque vous autorisez l'accès à l'aide d'un rôle IAM, Amazon Redshift crée automatiquement

des informations d'identification utilisateur temporaires à chaque séance. Pour plus d'informations, consultez Contrôle d'accès basé sur les rôles.

L'exemple suivant illustre la syntaxe du paramètre SESSION_TOKEN avec les paramètres ACCESS_KEY_ID et SECRET_ACCESS_KEY.

```
ACCESS_KEY_ID '<access-key-id>'
SECRET_ACCESS_KEY '<secret-access-key>'
SESSION_TOKEN '<temporary-token>';
```

Si vous spécifiez SESSION_TOKEN, vous ne pouvez utiliser ni IAM_ROLE, ni CREDENTIALS.

Utilisation du paramètre CREDENTIALS

CREDENTIALS

Clause qui indique la méthode que votre cluster utilisera pour accéder à d'autres AWS ressources contenant des fichiers de données ou des fichiers manifestes. Vous ne pouvez pas utiliser le paramètre CREDENTIALS avec IAM_ROLE ou ACCESS_KEY_ID et SECRET_ACCESS_KEY.

Voici la syntaxe du paramètre CREDENTIALS.

```
[WITH] CREDENTIALS [AS] 'credentials-args'
```

> **Note**
>
> Pour une meilleure flexibilité, nous vous recommandons d'utiliser le paramètre IAM_ROLE plutôt que le paramètre CREDENTIALS.

Le cas échéant, si le paramètre ENCRYPTED est utilisé, la chaîne credentials-args fournit également la clé de chiffrement.

La chaîne credentials-args est sensible à la casse et ne doit pas contenir d'espaces.

Les mots-clés WITH et AS sont facultatifs, et ils sont ignorés.

Vous pouvez spécifier role-based access control ou key-based access control. Dans les deux cas, le rôle IAM ou l'utilisateur doit avoir les autorisations nécessaires pour accéder aux ressources

AWS spécifiées. Pour de plus amples informations, veuillez consulter Autorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY.

> **Note**
>
> Pour protéger vos AWS informations d'identification et protéger les données sensibles, nous vous recommandons vivement d'utiliser un contrôle d'accès basé sur les rôles.

Pour spécifier un contrôle d'accès basé sur les rôles, fournissez la chaîne credentials-args au format suivant :

```
'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Pour utiliser les informations d'identification de jeton temporaires, vous devez fournir l'ID de clé d'accès temporaire, la clé d'accès secrète temporaire et le jeton temporaire. La chaîne credentials-args doit être au format suivant.

```
CREDENTIALS
'aws_access_key_id=<temporary-access-key-id>;aws_secret_access_key=<temporary-secret-access-key>;token=<temporary-token>'
```

Une commande COPY utilisant un contrôle d'accès basé sur les rôles avec des informations d'identification temporaires ressemblerait à l'exemple d'instruction suivant :

```
COPY customer FROM 's3://amzn-s3-demo-bucket/mydata'
CREDENTIALS
'aws_access_key_id=<temporary-access-key-id>;aws_secret_access_key=<temporary-secret-access-key-id>;token=<temporary-token>'
```

Pour de plus amples informations, veuillez consulter informations d'identification de sécurité temporaires.

Si le ENCRYPTED paramètre est utilisé, la chaîne credentials-args est au format suivant, où *<root-key>* est la valeur de la clé racine utilisée pour chiffrer les fichiers.

```
CREDENTIALS
'<credentials-args>;master_symmetric_key=<root-key>'
```

Une commande COPY utilisant un contrôle d'accès basé sur les rôles avec une clé de chiffrement ressemblerait à l'exemple d'instruction suivant :

```
COPY customer FROM 's3://amzn-s3-demo-bucket/mydata'
CREDENTIALS
'aws_iam_role=arn:aws:iam::<account-id>:role/<role-name>;master_symmetric_key=<root-key>'
```

Options de mappage de colonnes

Par défaut, la commande COPY insère des valeurs dans les colonnes de la table cible dans le même ordre que les champs se trouvent dans les fichiers de données. Si l'ordre des colonnes par défaut ne fonctionne pas, vous pouvez spécifier une liste de colonnes ou utiliser des JSONPath expressions pour mapper les champs de données source aux colonnes cibles.

- Column List
- JSONPaths File

Liste de colonnes

Vous pouvez spécifier une liste des noms de colonnes séparés par des virgules pour charger les champs de données source dans des colonnes cible spécifiques. Les colonnes peuvent être dans n'importe quel ordre dans l'instruction COPY, mais lors du chargement de fichiers plats, tels que dans un compartiment Amazon S3, leur ordre doit correspondre à celui des données sources.

Lors du chargement d'une table Amazon DynamoDB, l'ordre n'importe pas La commande COPY met en correspondance les noms d'attribut des éléments extraits de la table DynamoDB et les noms de colonne de la table Amazon Redshift. Pour plus d'informations, consultez Chargement de données à partir d'une table Amazon DynamoDB

Le format d'une liste de colonnes est le suivant.

```
COPY tablename (column1 [,column2, ...])
```

Si une colonne dans la table cible est absente de la liste de colonnes, la commande COPY charge l'expression DEFAULT de la colonne cible.

Si la colonne cible n'a pas de valeur par défaut, la commande COPY tente de charger NULL.

Si la commande COPY tente d'affecter NULL à une colonne qui est définie comme NOT NULL, elle échoue.

Si une colonne IDENTITY est incluse dans la liste de colonnes, EXPLICIT_IDS doit également être spécifié ; si une colonne IDENTITY n'est pas spécifiée, EXPLICIT_IDS ne peut pas être spécifié. Si aucune liste de la colonne n'est spécifiée, la commande se comporte comme si une liste de colonnes complète, dans l'ordre, a été spécifiée, avec les colonnes IDENTITY omises si EXPLICIT_IDS n'a pas non plus été spécifié.

Si une colonne est définie avec GENERATED BY DEFAULT AS IDENTITY, elle peut être copiée. Les valeurs sont générées ou mises à jour avec des valeurs que vous fournissez. L'option EXPLICIT_IDS n'est pas obligatoire. COPY ne met pas à jour le filigrane élevé d'identité. Pour de plus amples informations, veuillez consulter GENERATED BY DEFAULT AS IDENTITY.

JSONPaths fichier

Lors du chargement des fichiers de données au format JSON ou Avro, la commande COPY mappe automatiquement les éléments de données aux données source JSON ou Avro dans les colonnes de la table cible. Pour ce faire, elle met en correspondance les noms de champs du schéma Avro avec les noms de colonnes de la liste de tables ou de colonnes cible.

Dans certains cas, vos noms de colonnes et de champs ne correspondent pas, ou vous devez mapper à des niveaux plus profonds de la hiérarchie de données. Dans ces cas, vous pouvez utiliser un JSONPaths fichier pour mapper explicitement des éléments de données JSON ou Avro à des colonnes.

Pour de plus amples informations, veuillez consulter JSONPaths fichier.

Paramètres du format de données

Par défaut, la commande COPY attend que les données source soient dans un format texte UTF-8 séparé par des caractères. Le délimiteur par défaut est une barre verticale (|). Si les données source sont dans un autre format, utilisez les paramètres suivants pour spécifier le format des données:

- FORMAT
- CSV
- DELIMITER
- FIXEDWIDTH

- SHAPEFILE
- AVRO
- JSON
- PARQUET
- ORC

En plus des formats de données standard, COPY prend en charge les formats de données en colonnes suivants pour la commande COPY depuis Amazon S3 :

- ORC
- PARQUET

La commande COPY à partir du format en colonne est pris en charge avec certaines restrictions. Pour plus d'informations, consultez COPY depuis les formats de données en colonnes.

Paramètres du format de données

FORMAT [AS]

(Facultatif) Identifie les mots-clés de format des données. Les arguments FORMAT sont décrits ci-après.

CSV [QUOTE [AS] 'quote_character']

Permet l'utilisation du format CSV dans les données d'entrée. Pour insérer automatiquement des délimiteurs, des caractères de saut de ligne et des retours à la ligne, entourez le champ dans le caractère spécifié par le paramètre QUOTE. Le guillemet par défaut correspond aux guillemets doubles ("). Lorsque le guillemet simple est utilisé dans un champ, précédez le caractère d'un guillemet simple supplémentaire. Par exemple, si les guillemets sont doubles, pour insérer la chaîne `A "quoted" word` le fichier d'entrée doit inclure la chaîne `"A ""quoted"" word"`. Lorsque le paramètre CSV est utilisé, le délimiteur par défaut est une virgule (,). Vous pouvez spécifier un autre délimiteur en utilisant le paramètre DELIMITER.

Lorsqu'un champ est placé entre guillemets, l'espace blanc entre les délimiteurs et les guillemets est ignoré. Si le délimiteur est un caractère d'espace vide, comme un onglet, le délimiteur n'est pas considéré comme un espace vide.

CSV ne peut pas être utilisé avec FIXEDWIDTH, REMOVEQUOTES ou ESCAPE.

QUOTE [AS] ’quote_character’

Facultatif. Spécifie le caractère à utiliser comme guillemet lorsque vous utilisez le paramètre CSV. La valeur par défaut est les guillemets doubles ("). Si vous utilisez le paramètre QUOTE pour définir un autre guillemet que des guillemets doubles, vous n’avez pas besoin d’insérer les guillemets doubles dans le champ. Le paramètre QUOTE peut être utilisé uniquement avec le paramètre CSV. Le mot-clé AS est facultatif.

DELIMITER [AS] [’delimiter_char’]

Spécifie les caractères utilisés pour séparer les champs du fichier d'entrée, tels qu'un tube (`|`), une virgule (`,`), un onglet (`\t`) ou plusieurs caractères tels que `|~|`. Les caractères non imprimables sont pris en charge. Les caractères peuvent également être représentés en octal sous forme d'unités de code UTF-8. Pour l'octal, utilisez le format « \ ddd », où « d » est un chiffre octal (0—7). Le séparateur par défaut est un caractère tube (`|`), sauf si le paramètre CSV est utilisé, auquel cas le délimiteur par défaut est une virgule (). `,` Le mot-clé AS est facultatif. DELIMITER ne peut pas être utilisé avec FIXEDWIDTH.

FIXEDWIDTH ’fixedwidth_spec’

Charge les données d’un fichier où la largeur de chaque colonne est une longueur fixe, plutôt que des colonnes séparées par un délimiteur. fixedwidth_spec est une chaîne qui spécifie une étiquette de colonne définie par l’utilisateur et la largeur de la colonne. L’étiquette de colonne peut être une chaîne de texte ou un nombre entier, en fonction de ce que l’utilisateur choisit. L’étiquette de colonne n’a aucun lien avec le nom de la colonne. L'ordre des label/width paires doit correspondre exactement à l'ordre des colonnes du tableau. FIXEDWIDTH ne peut pas être utilisé avec CSV ou DELIMITER. Dans Amazon Redshift la longueur des colonnes CHAR et VARCHAR est exprimée en octets, aussi n’oubliez pas que la largeur de colonne que vous spécifiez correspond à la longueur binaire de caractères sur plusieurs octets lors de la préparation du fichier à charger. Pour plus d'informations, consultez Types caractères.

Le format de fixedwidth_spec apparaît comme suit :

```
'colLabel1:colWidth1,colLabel:colWidth2, ...'
```

SHAPEFILE [SIMPLIFY [AUTO] [’tolerance’]]

Permet l’utilisation du format CSV dans les données d’entrée. Par défaut, la première colonne du shapefile est une colonne `GEOMETRY` ou `IDENTITY`. Toutes les colonnes suivantes suivent l’ordre spécifié dans le shapefile.

Vous ne pouvez pas utiliser SHAPEFILE avec FIXEDWIDTH, REMOVEQUOTES ou ESCAPE.

Pour utiliser des objets `GEOGRAPHY` avec `COPY FROM SHAPEFILE`, intégrez-les d'abord dans une colonne `GEOMETRY`, puis convertissez-les en objets `GEOGRAPHY`.

SIMPLIFY [tolerance]

(Facultatif) Simplifie toutes les géométries pendant le processus d'ingestion à l'aide de l' Ramer-Douglas-Peuckeralgorithme et de la tolérance donnée.

SIMPLIFY AUTO [tolerance]

(Facultatif) Simplifie uniquement les géométries supérieures à la taille de géométrie maximale. Cette simplification utilise l' Ramer-Douglas-Peuckeralgorithme et la tolérance calculée automatiquement si celle-ci ne dépasse pas la tolérance spécifiée. Cet algorithme calcule la taille pour stocker les objets dans la tolérance spécifiée. La valeur tolerance est facultative.

Pour voir des exemples de chargement de shapefiles, consultez Chargement d'un shapefile dans Amazon Redshift.

AVRO [AS] 'avro_option'

Spécifie que les données source sont au format Avro.

Le format Avro est pris en charge pour la commande COPY dans les services et protocoles suivants :

- Amazon S3
- Amazon EMR
- Hôtes distants (SSH)

Avro n'est pas pris en charge pour la commande COPY dans DynamoDB.

Avro est un protocole de sérialisation de données. Un fichier source Avro inclut un schéma qui définit la structure des données. Le type de schéma Avro doit être `record`. La commande COPY accepte la création de fichiers Avro à l'aide du codec non compressé par défaut, ainsi que les codecs de compression `deflate` et `snappy`. Pour plus d'informations sur Avro, accédez à Apache Avro.

Les valeurs valides pour avro_option sont les suivantes :

- `'auto'`
- `'auto ignorecase'`
- `'s3://jsonpaths_file'`

La valeur par défaut est `'auto'`.

COPY mappe automatiquement les éléments de données des données source Avro aux colonnes de la table cible en mettant en correspondance les noms de champs du schéma Avro avec les noms de colonnes dans la table cible. La mise en correspondance est sensible à la casse pour `'auto'` et n'est pas sensible à la casse pour `'auto ignorecase'`.

Les noms de colonnes des tables Amazon Redshift sont toujours en lettres minuscules, aussi lorsque vous utilisez l'option `'auto'`, les noms de champs correspondants doivent être en minuscules. Si les noms de champs ne sont pas tous en minuscules, vous pouvez utiliser l'option `'auto ignorecase'`. Avec l'argument `'auto'` par défaut, la commande COPY ne reconnaît que le premier niveau de champs (ou champs externes) dans la structure.

Pour mapper explicitement les noms de colonnes aux noms de champs Avro, vous pouvez utiliser JSONPaths fichier.

Par défaut, la commande COPY tente de mettre en correspondance toutes les colonnes de la table cible avec les noms de champs Avro. Pour charger un sous-ensemble de colonnes, vous pouvez éventuellement spécifier une liste de colonnes. Si une colonne dans la table cible est absente de la liste de colonnes, la commande COPY charge l'expression DEFAULT de la colonne cible. Si la colonne cible n'a pas de valeur par défaut, la commande COPY tente de charger NULL. Si une colonne est incluse dans la liste de colonnes et que la commande COPY ne trouve pas de champ correspondant dans les données Avro, la commande COPY tente de charger NULL dans la colonne.

Si la commande COPY tente d'affecter NULL à une colonne qui est définie comme NOT NULL, elle échoue.

Schéma Avro

Un fichier de données source Avro inclut un schéma qui définit la structure des données. La commande COPY lit le schéma qui fait partie du fichier de données source Avro pour mapper des éléments de données aux colonnes de la table cible. L'exemple suivant illustre un schéma Avro.

```
{
    "name": "person",
    "type": "record",
    "fields": [
        {"name": "id", "type": "int"},
        {"name": "guid", "type": "string"},
        {"name": "name", "type": "string"},
```

```
        {"name": "address", "type": "string"}]
}
```

Le schéma Avro est défini à l'aide du format JSON. L'objet de niveau supérieur JSON contient trois paires nom-valeur avec les noms ou clés, `"name"`, `"type"` et `"fields"`.

La clé `"fields"` s'apparie à un tableau d'objets qui définissent le nom et le type de données de chaque champ dans la structure de données. Par défaut, la commande COPY met automatiquement en correspondance les noms de champs avec les noms de colonnes. Les noms de colonnes sont toujours en minuscules, les noms de champs correspondants doivent donc toujours être en minuscules, sauf si vous spécifiez l'option `'auto ignorecase'`. Tous les noms de domaines qui ne correspondent pas à un nom de colonne sont ignorés. L'ordre n'a pas d'importance. Dans l'exemple précédent, la commande COPY mappe aux noms de colonnes `id`, `guid`, `name`, et `address`.

Avec l'argument `'auto'` par défaut, la commande COPY met en correspondance uniquement les objets de premier niveau des colonnes. Pour effectuer un mappage vers des niveaux plus profonds du schéma, ou si les noms de champs et de colonnes ne correspondent pas, utilisez un JSONPaths fichier pour définir le mappage. Pour de plus amples informations, veuillez consulter JSONPaths fichier.

Si la valeur associée à une clé est un type de données complexe Avro comme octets, tableau, enregistrement, carte ou lien, la commande COPY charge la valeur sous forme de chaîne. Ici, la chaîne est la représentation JSON des données. La commande COPY charge les types de données d'énumération Avro sous forme de chaînes dans lesquelles le contenu est le nom du type. Pour obtenir un exemple, consultez Exécution de la commande COPY à partir du format JSON.

La taille maximale de l'en-tête du fichier Avro, qui inclut les métadonnées de fichiers et de schéma, est de 1 Mo.

La taille maximale d'un seul bloc de données Avro est de 4 Mo. Ce qui est différent de la taille de ligne maximale. Si la taille maximale d'un seul bloc de données Avro est dépassée, même si la taille de la ligne qui en résulte est inférieure à la limite de taille de ligne de 4 Mo, la commande COPY échoue.

Lors du calcul de la taille des lignes, Amazon Redshift compte deux fois les barres verticales (|) en interne. Si les données entrées contiennent un très grand nombre de barres verticales, la taille des lignes peut dépasser 4 Mo même si la taille du bloc de données est inférieure à 4 Mo.

JSON [AS] 'json_option'

Les données source sont au format JSON.

Le format JSON est pris en charge pour la commande COPY dans ces services et protocoles :

- Amazon S3
- Commande COPY depuis Amazon EMR
- Exécution de la commande COPY depuis SSH

JSON n'est pas pris en charge pour exécuter la commande COPY dans DynamoDB.

Les valeurs valides pour json_option sont les suivantes :

- `'auto'`
- `'auto ignorecase'`
- `'s3://jsonpaths_file'`
- `'noshred'`

La valeur par défaut est `'auto'`. Amazon Redshift ne déchiquète pas les attributs des structures JSON en plusieurs colonnes lors du chargement d'un document JSON.

Par défaut, la commande COPY tente de mettre en correspondance toutes les colonnes de la table cible avec les clés de noms de champs JSON. Pour charger un sous-ensemble de colonnes, vous pouvez éventuellement spécifier une liste de colonnes. Si les clés de nom de domaine JSON ne sont pas toutes en minuscules, vous pouvez utiliser l'option `'auto ignorecase'` ou JSONPaths fichier pour mapper explicitement les noms de colonnes aux clés de nom de domaine JSON.

Si une colonne dans la table cible est absente de la liste de colonnes, la commande COPY charge l'expression DEFAULT de la colonne cible. Si la colonne cible n'a pas de valeur par défaut, la commande COPY tente de charger NULL. Si une colonne est incluse dans la liste de colonnes et que la commande COPY ne trouve pas de champ correspondant dans les données JSON, la commande COPY tente de charger NULL dans la colonne.

Si la commande COPY tente d'affecter NULL à une colonne qui est définie comme NOT NULL, elle échoue.

La commande COPY mappe les éléments de données dans les données source JSON aux colonnes de la table cible. Pour cela, elle fait correspondre les clés d'objet (ou les noms) dans les paires nom-valeur source aux noms des colonnes de la table cible.

Reportez-vous aux informations suivantes sur chaque valeur json_option :

’auto’

Avec cette option, la mise en correspondance est sensible à la casse. Les noms de colonnes des tables Amazon Redshift sont toujours en lettres minuscules, aussi lorsque vous utilisez l’option `'auto'`, les noms de champs JSON correspondants doivent être en minuscules.

’auto ignorecase’

Avec cette option, la mise en correspondance n’est pas sensible à la casse. Les noms de colonnes des tables Amazon Redshift étant toujours en minuscules, lorsque vous utilisez la commande `'auto ignorecase'`, les noms de champs JSON correspondants peuvent être en minuscules, en majuscules ou en casse mixte.

’s3://jsonpaths_file’

Avec cette option, COPY utilise le JSONPaths fichier nommé pour mapper les éléments de données des données sources JSON aux colonnes de la table cible. L’argument *`s3://jsonpaths_file`* doit être une clé d’objet Amazon S3 faisant explicitement référence à un seul fichier. Par exemple : `'s3://amzn-s3-demo-bucket/jsonpaths.txt`’. L’argument ne peut pas être un préfixe de clé. Pour plus d'informations sur l'utilisation d'un JSONPaths fichier, consultezthe section called “JSONPaths fichier”.

Dans certains cas, le fichier spécifié par `jsonpaths_file` a le même préfixe que le chemin spécifié par `copy_from_s3_objectpath` pour les fichiers de données. Dans ce cas, COPY lit le JSONPaths fichier sous forme de fichier de données et renvoie des erreurs. Supposons, par exemple, que vos fichiers de données utilisent le chemin de l'objet `s3://amzn-s3-demo-bucket/my_data.json` et que votre JSONPaths fichier le soits`3://amzn-s3-demo-bucket/my_data.jsonpaths`. Dans ce cas, la commande COPY tente de charger `my_data.jsonpaths` en tant que fichier de données.

’noshred’

Avec cette option, Amazon Redshift ne déchiquète pas les attributs des structures JSON en plusieurs colonnes lors du chargement d’un document JSON.

Fichier de données JSON

Le fichier de données JSON contient un ensemble d’objets ou tableaux. La commande COPY charge chaque objet ou tableau JSON dans une seule ligne de la table cible. Chaque objet ou tableau

correspondant à une ligne doit être une structure autonome, de niveau racine, autrement dit, il ne doit être membre d'une autre structure JSON.

Un objet JSON commence et finit par des accolades ({ }) et contient une collection non ordonnée de paires nom-valeur. Chaque paire de nom et de valeur est séparée par deux points, et les paires sont séparées par des virgules. Par défaut, la clé d'objet, ou le nom, dans les paires nom-valeur, doit correspondre au nom de la colonne correspondante dans la table. Les noms de colonnes des tables Amazon Redshift étant toujours en minuscules, les clés des noms de champs JSON correspondants doivent également être en minuscules. Si vos noms de colonnes et clés JSON ne correspondent pas, utilisez un the section called "JSONPaths fichier" pour mapper explicitement les colonnes aux clés.

L'ordre d'un objet JSON n'est pas important. Tous les noms qui ne correspondent pas à un nom de colonne sont ignorés. L'exemple suivant illustre la structure d'un objet JSON simple.

```
{
  "column1": "value1",
  "column2": value2,
  "notacolumn" : "ignore this value"
}
```

Un tableau JSON commence et finit par des crochets ([ ]) et contient une collection ordonnée de valeurs séparées par des virgules. Si vos fichiers de données utilisent des tableaux, vous devez spécifier un JSONPaths fichier pour faire correspondre les valeurs aux colonnes. L'exemple suivant illustre la structure d'un tableau JSON simple.

```
["value1", value2]
```

Le JSON doit être dans un format correct. Par exemple, les objets ou les tableaux ne peuvent pas être séparés par des virgules ou tout autre caractère à l'exception des espaces vides. Les chaînes doivent être placées entre guillemets doubles. Les guillemets doivent être des guillemets simples (0x22), ni culbutés, ni courbes.

La taille maximale d'un seul objet ou tableau JSON, y compris les accolades ou les crochets, est de 4 Mo. Ce qui est différent de la taille de ligne maximale. Si la taille maximale d'un seul objet ou tableau JSON est dépassée, même si la taille de la ligne qui en résulte est inférieure à la limite de taille de ligne de 4 Mo, la commande COPY échoue.

Lors du calcul de la taille des lignes, Amazon Redshift compte deux fois les barres verticales (|) en interne. Si les données entrées contiennent un très grand nombre de barres verticales, la taille des lignes peut dépasser 4 Mo même si la taille de l'objet est inférieure à 4 Mo.

La commande COPY charge \n comme un caractère de saut de ligne et charge \t comme un caractère de tabulation. Pour charger une barre oblique inverse, précédez-la d'une barre oblique inverse (\\ ).

La commande COPY recherche la source JSON spécifiée pour un objet ou tableau JSON bien formé et valide. Si la commande COPY rencontre des caractères qui ne sont pas des espaces vides avant de localiser une structure JSON utilisable, ou entre des objets ou tableaux JSON valides, elle renvoie une erreur pour chaque instance. Ces erreurs sont prises en compte dans le nombre d'erreurs MAXERROR. Lorsque le nombre d'erreurs est égal ou supérieur à MAXERROR, la commande COPY échoue.

Pour chaque erreur, Amazon Redshift enregistre une ligne dans la table système STL_LOAD_ERRORS. La colonne LINE_NUMBER enregistre la dernière ligne de l'objet JSON qui a provoqué l'erreur.

Si IGNOREHEADER est spécifié, la commande COPY ignore le nombre de lignes dans les données JSON spécifiées. Les caractères de saut de ligne dans les données JSON comptent toujours pour les calculs IGNOREHEADER.

La commande COPY charge des chaînes vides dans les champs vides par défaut. Si EMPTYASNULL est spécifié, la commande COPY charge des chaînes vides dans les champs CHAR et VARCHAR comme NULL. Les chaînes vides d'autres types de données, tels que INT, sont toujours chargées avec NULL.

Les options suivantes ne sont pas prises en charge avec JSON :

- CSV
- DELIMITER
- ESCAPE
- FILLRECORD
- FIXEDWIDTH
- IGNOREBLANKLINES
- NULL AS
- READRATIO
- REMOVEQUOTES

Pour plus d'informations, consultez Exécution de la commande COPY à partir du format JSON. Pour plus d'informations sur les structures de données JSON, accédez à www.json.org.

JSONPaths fichier

Si vous chargez depuis des données source au format JSON ou Avro, la commande COPY mappe par défaut les éléments de données de premier niveau dans les données source aux colonnes de la table cible. Pour ce faire, elle met en correspondance chaque nom (ou clé d'objet) dans une paire nom-valeur avec le nom d'une colonne de la table cible.

Si les noms de vos colonnes et vos clés d'objet ne correspondent pas, ou pour les mapper à des niveaux plus profonds de la hiérarchie des données, vous pouvez utiliser un JSONPaths fichier pour mapper explicitement des éléments de données JSON ou Avro à des colonnes. Le JSONPaths fichier mappe les éléments de données JSON aux colonnes en faisant correspondre l'ordre des colonnes dans la table ou la liste de colonnes cible.

Le JSONPaths fichier ne doit contenir qu'un seul objet JSON (pas un tableau). L'objet JSON est une paire nom-valeur. La clé d'objet, qui est le nom de la paire nom-valeur, doit être `"jsonpaths"`. La valeur de la paire nom-valeur est un tableau d'JSONPath expressions. Chaque JSONPath expression fait référence à un seul élément de la hiérarchie de données JSON ou du schéma Avro, de la même manière qu'une XPath expression fait référence aux éléments d'un document XML. Pour de plus amples informations, veuillez consulter JSONPath expressions.

Pour utiliser un JSONPaths fichier, ajoutez le mot clé JSON ou AVRO à la commande COPY. Spécifiez le nom du compartiment S3 et le chemin de l'objet du JSONPaths fichier en utilisant le format suivant.

```
COPY tablename
FROM 'data_source'
CREDENTIALS 'credentials-args'
FORMAT AS { AVRO | JSON } 's3://jsonpaths_file';
```

La valeur `s3://jsonpaths_file` doit être une clé d'objet Amazon S3 faisant explicitement référence à un seul fichier, tel que `'s3://amzn-s3-demo-bucket/jsonpaths.txt'`. Elle ne pas être un préfixe de clé.

Dans certains cas, si vous chargez à partir d'Amazon S3, le fichier spécifié par `jsonpaths_file` a le même préfixe que le chemin spécifié par `copy_from_s3_objectpath` pour les fichiers de données. Dans ce cas, COPY lit le JSONPaths fichier sous forme de fichier de données et renvoie

des erreurs. Supposons, par exemple, que vos fichiers de données utilisent le chemin de l'objet `s3://amzn-s3-demo-bucket/my_data.json` et que votre JSONPaths fichier le soits3://amzn-s3-demo-bucket/my_data.jsonpaths. Dans ce cas, la commande COPY tente de charger `my_data.jsonpaths` en tant que fichier de données.

Si le nom de clé est une chaîne autre que `"jsonpaths"`, la commande COPY ne renvoie pas d'erreur, mais elle ignore jsonpaths_file et utilise l'argument `'auto'` à la place.

Si l'une des actions suivantes se produit, la commande COPY échoue :

- Le JSON est incorrect.
- Il existe plusieurs objets JSON.
- Tous les caractères sauf les espaces vides existent en dehors de l'objet.
- Un élément de tableau est une chaîne vide ou n'est pas une chaîne.

MAXERROR ne s'applique pas au JSONPaths fichier.

Le JSONPaths fichier ne doit pas être chiffré, même si l'ENCRYPTEDoption est spécifiée.

Pour de plus amples informations, veuillez consulter Exécution de la commande COPY à partir du format JSON.

JSONPath expressions

Le JSONPaths fichier utilise des JSONPath expressions pour mapper les champs de données aux colonnes cibles. Chaque JSONPath expression correspond à une colonne de la table cible Amazon Redshift. L'ordre des éléments du JSONPath tableau doit correspondre à l'ordre des colonnes de la table cible ou de la liste des colonnes, si une liste de colonnes est utilisée.

Les guillemets doubles sont obligatoires comme illustré, aussi bien pour les noms de champ que pour les valeurs. Les guillemets doivent être des guillemets simples (0x22), ni culbutés, ni courbes.

Si aucun élément d'objet référencé par une JSONPath expression n'est trouvé dans les données JSON, COPY tente de charger une valeur NULL. Si l'objet référencé est incorrect, la commande COPY renvoie une erreur de chargement.

Si aucun élément de tableau référencé par une JSONPath expression n'est trouvé dans les données JSON ou Avro, COPY échoue avec l'erreur suivante : `Invalid JSONPath format: Not an`

`array or index out of range.` Supprimez tous les éléments du tableau JSONPaths qui n'existent pas dans les données sources et vérifiez que les tableaux des données sources sont bien formés.

Les JSONPath expressions peuvent utiliser la notation entre crochets ou la notation par points, mais vous ne pouvez pas mélanger les notations. L'exemple suivant montre des JSONPath expressions utilisant la notation entre crochets.

```
{
    "jsonpaths": [
        "$['venuename']",
        "$['venuecity']",
        "$['venuestate']",
        "$['venueseats']"
    ]
}
```

L'exemple suivant montre des JSONPath expressions utilisant la notation par points.

```
{
    "jsonpaths": [
        "$.venuename",
        "$.venuecity",
        "$.venuestate",
        "$.venueseats"
    ]
}
```

Dans le contexte de la syntaxe COPY d'Amazon Redshift, une JSONPath expression doit spécifier le chemin explicite vers un élément de nom unique dans une structure de données hiérarchique JSON ou Avro. Amazon Redshift ne prend en charge aucun JSONPath élément, tel que les caractères génériques ou les expressions de filtre, susceptibles de donner lieu à un chemin ambigu ou à plusieurs éléments de nom.

Pour de plus amples informations, veuillez consulter Exécution de la commande COPY à partir du format JSON.

Utilisation JSONPaths avec Avro Data

L’exemple suivant illustre un schéma Avro à plusieurs niveaux.

```
{
    "name": "person",
    "type": "record",
    "fields": [
        {"name": "id", "type": "int"},
        {"name": "guid", "type": "string"},
        {"name": "isActive", "type": "boolean"},
        {"name": "age", "type": "int"},
        {"name": "name", "type": "string"},
        {"name": "address", "type": "string"},
        {"name": "latitude", "type": "double"},
        {"name": "longitude", "type": "double"},
        {
            "name": "tags",
            "type": {
                        "type" : "array",
                        "name" : "inner_tags",
                        "items" : "string"
                    }
        },
        {
            "name": "friends",
            "type": {
                        "type" : "array",
                        "name" : "inner_friends",
                        "items" : {
                                    "name" : "friends_record",
                                    "type" : "record",
                                    "fields" : [
                                                {"name" : "id", "type" : "int"},
                                                {"name" : "name", "type" : "string"}
                                               ]
                                  }
                    }
        },
        {"name": "randomArrayItem", "type": "string"}
    ]
}
```

L'exemple suivant montre un JSONPaths fichier qui utilise des AvroPath expressions pour faire référence au schéma précédent.

```
{
```

```
    "jsonpaths": [
        "$.id",
        "$.guid",
        "$.address",
        "$.friends[0].id"
    ]
}
```

L' JSONPaths exemple inclut les éléments suivants :

jsonpaths

Nom de l'objet JSON qui contient les AvroPath expressions.

[…]

Les crochets entourent le tableau JSON qui contient les éléments de chemin d’accès.

$

Le symbole du dollar fait référence à l’élément racine du schéma Avro, qui est le tableau `"fields"`.

"$.id",

La cible de l' AvroPath expression. Dans ce cas, la cible est l’élément dans le tableau `"fields"` avec le nom `"id"`. Les expressions sont séparées par des virgules.

"$.friends[0].id"

Les crochets indiquent un index de tableau. JSONPath les expressions utilisent une indexation basée sur zéro. Cette expression fait donc référence au premier élément du `"friends"` tableau portant le nom. `"id"`

La syntaxe du schéma Avro nécessite l’utilisation de champs internes pour définir la structure d’enregistrement et les types de données du tableau. Les champs internes sont ignorés par les AvroPath expressions. Par exemple, le champ `"friends"` définit un tableau nommé `"inner_friends"`, qui définit à son tour un enregistrement nommé `"friends_record"`. L' AvroPath expression faisant référence au champ `"id"` peut ignorer les champs supplémentaires pour référencer directement le champ cible. Les AvroPath expressions suivantes font référence aux deux champs appartenant au `"friends"` tableau.

```
"$.friends[0].id"
```

```
"$.friends[0].name"
```

Paramètres du format des données en colonnes

En plus des formats de données standard, COPY prend en charge les formats de données en colonnes suivants pour la commande COPY depuis Amazon S3. La commande COPY à partir du format en colonne est pris en charge avec certaines restrictions. Pour plus d'informations, consultez COPY depuis les formats de données en colonnes.

ORC

Charge les données à partir d'un fichier qui utilise le format de fichier ORC (Optimized Row Columnar).

PARQUET

Charge les données à partir d'un fichier qui utilise le format de fichier Parquet.

Paramètres de compression de fichier

Vous pouvez charger à partir de fichiers de données compressés en spécifiant les paramètres suivants.

Paramètres de compression de fichier

BZIP2

Valeur qui indique que le ou les fichiers d'entrée sont au format compressé bzip2 (fichiers .bz2). L'opération COPY lit chaque fichier compressé et décompresse les données qu'elle charge.

GZIP

Valeur qui indique que le ou les fichiers d'entrée sont au format compressé gzip (fichiers .gz). L'opération COPY lit chaque fichier compressé et décompresse les données qu'elle charge.

LZOP

Valeur qui indique que le ou les fichiers d'entrée sont au format compressé lzop (fichiers .lzo). L'opération COPY lit chaque fichier compressé et décompresse les données qu'elle charge.

> **Note**
>
> L'opération COPY ne prend pas en charge les fichiers qui sont compressés à l'aide de l'option lzop --filter.

ZSTD

Valeur qui indique que le ou les fichiers d'entrée sont au format compressé Zstandard (fichiers .zst). L'opération COPY lit chaque fichier compressé et décompresse les données qu'elle charge.

> **Note**
>
> ZSTD est pris en charge uniquement avec la commande COPY depuis Amazon S3.

Paramètres de conversion de données

Lorsqu'elle charge la table, la commande COPY tente implicitement de convertir les chaînes dans les données source vers le type de données de la colonne cible. Si vous devez spécifier une conversion qui est différente du comportement par défaut, ou si la conversion par défaut entraîne des erreurs, vous pouvez gérer les conversions de données en spécifiant les paramètres suivants. Pour plus d'informations sur la syntaxe de ces paramètres, consultez Syntaxe de la commande COPY.

- ACCEPTANYDATE
- ACCEPTINVCHARS
- BLANKSASNULL
- DATEFORMAT
- EMPTYASNULL
- ENCODING
- ESCAPE
- EXPLICIT_IDS
- FILLRECORD
- IGNOREBLANKLINES
- IGNOREHEADER

- NULL AS
- REMOVEQUOTES
- ROUNDEC
- TIMEFORMAT
- TRIMBLANKS
- TRUNCATECOLUMNS

Paramètres de conversion de données

ACCEPTANYDATE

Autorise n'importe quel format de date, y compris les formats non valides comme `00/00/00 00:00:00`, à charger sans générer d'erreur. Ce paramètre s'applique uniquement aux colonnes TIMESTAMP et DATE. Utilisez toujours ACCEPTANYDATE avec le paramètre DATEFORMAT. Si le format de date pour les données ne correspond pas à la spécification DATEFORMAT, Amazon Redshift insère une valeur NULL dans ce champ.

ACCEPTINVCHARS [AS] ['replacement_char']

Autorise le chargement de données dans les colonnes VARCHAR, même si les données contiennent des caractères UTF-8 non valides. Lorsque ACCEPTINVCHARS est spécifié, la commande COPY remplace chaque caractère UTF-8 non valide par une chaîne de la même longueur comprenant le caractère spécifié par replacement_char. Par exemple, si le caractère de remplacement est '^', un caractère non valide de trois octets sera remplacé par '^^^'.

Le caractère de remplacement peut être n'importe quel caractère ASCII sauf NULL. La valeur par défaut est un point d'interrogation (?). Pour plus d'informations sur les caractères UTF-8 non valides, consultez Erreurs de chargement de caractères multioctets.

La commande COPY renvoie le nombre de lignes qui contenaient des caractères UTF-8 non valides, et ajoute une entrée à la table système STL_REPLACEMENTS pour chaque ligne concernée, à concurrence de 100 lignes au maximum pour chaque tranche de nœud. Les caractères UTF-8 non valides supplémentaires sont également remplacés, mais ces événements de remplacement ne sont pas enregistrés.

Si ACCEPTINVCHARS n'est pas spécifié, la commande COPY renvoie une erreur chaque fois qu'elle rencontre un caractère UTF-8 non valide.

ACCEPTINVCHARS est valide uniquement pour les colonnes VARCHAR.

BLANKSASNULL

Charge les champs vides, qui comprennent des espaces vides uniquement, comme NULL. Cette option s'applique uniquement aux colonnes CHAR et VARCHAR. Les champs vides pour d'autres types de données, tels que INT, sont toujours chargés avec NULL. Par exemple, une chaîne qui contient trois espaces à la suite (et aucun autre caractère) est chargée comme une valeur NULL. Le comportement par défaut, sans cette option, consiste à charger les espaces tels quels.

DATEFORMAT [AS] {'dateformat_string' | 'auto' }

Si aucun DATEFORMAT n'est spécifié, le format par défaut est `'YYYY-MM-DD'`. Par exemple, un autre format valide est `'MM-DD-YYYY'`.

Si la commande COPY ne reconnaît pas le format de vos valeurs de date ou d'heure, ou si vos valeurs de date ou d'heure utilisent des formats différents, utilisez l'argument `'auto'` avec le paramètre DATEFORMAT ou TIMEFORMAT. L'argument `'auto'` reconnaît plusieurs formats qui ne sont pas pris en charge lors de l'utilisation d'une chaîne DATEFORMAT et TIMEFORMAT. Le mot-clé `'auto'` est sensible à la casse. Pour plus d'informations, consultez Utilisation de la reconnaissance automatique avec DATEFORMAT et TIMEFORMAT.

Le format de date peut inclure des informations de temps (heure, minutes, secondes), mais ces informations sont ignorées. Le mot-clé AS est facultatif. Pour plus d'informations, consultez Chaînes DATEFORMAT et TIMEFORMAT.

EMPTYASNULL

Indique que Amazon Redshift doit charger les champs CHAR et VARCHAR vides comme des valeurs NULL. Les champs vides pour d'autres types de données, tels que INT, sont toujours chargés avec NULL. Les champs vides surviennent lorsque les données contiennent deux délimiteurs de suite sans caractère entre les délimiteurs. EMPTYASNULL et NULL AS '' (chaîne vide) entraînent le même comportement.

ENCODING [AS] file_encoding

Spécifie le type d'encodage des données de chargement. La commande COPY convertit les données de l'encodage spécifié dans UTF-8 pendant le chargement.

Les valeurs valides pour file_encoding sont les suivantes :

- `UTF8`
- `UTF16`
- `UTF16LE`

- UTF16BE
- ISO88591

La valeur par défaut est UTF8.

Les noms de fichier source doivent utiliser l'encodage UTF-8.

Les fichiers suivants doivent utiliser l'encodage UTF-8, même si un autre encodage a été spécifié pour les données de chargement :

- Fichiers manifestes
- JSONPaths fichiers

Les chaînes d'arguments fournies avec les paramètres suivants doivent utiliser UTF-8 :

- FIXEDWIDTH 'fixedwidth_spec'
- ACCEPTINVCHARS 'replacement_char'
- DATEFORMAT 'dateformat_string'
- TIMEFORMAT 'timeformat_string'
- NULL AS 'null_string'

Les fichiers de données de largeur fixe doivent utiliser l'encodage UTF-8. Les largeurs des champs dépendent du nombre de caractères, et non du nombre d'octets.

Toutes les données de chargement doivent utiliser l'encodage spécifié. Si la commande COPY rencontre un autre encodage, elle ignore le fichier et renvoie une erreur.

Si vous spécifiez UTF16, vos données doivent comporter une marque d'ordre d'octet. Si vous savez que vos données UTF-16 sont de poids faible (Little endian) ou de poids fort (Big endian), vous pouvez utiliser UTF16LE ou UTF16BE, indépendamment de la présence d'une marque d'ordre d'octet.

Pour utiliser le codage ISO-8859-1, spécifiez. ISO88591 Pour plus d'informations, consultez la norme ISO/IEC 8859-1 sur Wikipedia.

ESCAPE

Lorsque ce paramètre est spécifié, la barre oblique inverse (\) dans les données d'entrée est considérée comme un caractère d'échappement. Le caractère qui se trouve immédiatement après la barre oblique inverse est chargé dans la table comme faisant partie de la valeur de la colonne actuelle, même s'il s'agit d'un caractère qui a normalement une fonction particulière. Par exemple, vous pouvez utiliser ce paramètre pour insérer le caractère délimiteur, un guillemet anglais, un

caractère de saut de ligne intégré, ou le caractère d'échappement lui-même lorsque l'un de ces caractères fait légitimement partie d'une valeur de la colonne.

Si vous spécifiez le paramètre ESCAPE en liaison avec le paramètre REMOVEQUOTES, vous pouvez insérer et conserver des guillemets doubles (`'` ou `''`) qui peuvent être supprimés dans le cas contraire. La chaîne vide par défaut, `\N`, fonctionne telle quelle, mais elle peut également être insérée dans les données d'entrée en tant que `\\N`. Tant que vous ne spécifiez pas d'autre chaîne nulle avec le paramètre NULL AS, `\N` et `\\N` entraînent les mêmes résultats.

> **Note**
>
> Le caractère de contrôle `0x00` (NUL) ne peut pas être inséré et doit être supprimé des données d'entrée ou converti. Ce caractère est considéré comme une marque de fin d'enregistrement, ce qui entraîne la troncation du reste de l'enregistrement.

Vous ne pouvez pas utiliser le paramètre ESCAPE pour les charges FIXEDWIDTH, et vous ne pouvez pas spécifier le caractère d'échappement lui-même ; il s'agit toujours de la barre oblique inverse. En outre, vous devez vous assurer que les données d'entrée contiennent le caractère d'échappement aux endroits appropriés.

Voici des exemples de données d'entrée et des données chargées résultant de celles-ci lorsque le paramètre ESCAPE est spécifié. Le résultat de la ligne 4 part du principe que le paramètre REMOVEQUOTES est également spécifié. Les données d'entrée se composent de deux champs séparés par une barre verticale :

```
1|The quick brown fox\[newline]
jumped over the lazy dog.
2| A\\B\\C
3| A \| B \| C
4| 'A Midsummer Night\'s Dream'
```

Les données chargées dans la colonne 2 ressemblent à ceci :

```
The quick brown fox
jumped over the lazy dog.
A\B\C
A|B|C
A Midsummer Night's Dream
```

> **Note**
>
> L'application du caractère d'échappement aux données d'entrée pour une charge est de la responsabilité de l'utilisateur, sauf lorsque vous rechargez les données qui ont été précédemment déchargées avec le paramètre ESCAPE. Dans ce cas, les données contiennent déjà les caractères d'échappement nécessaires.

Le paramètre ESCAPE n'interprète pas la notation de séquence d'échappement octale, hexadécimale, Unicode, ou autre. Par exemple, si vos données source contiennent la valeur de saut de ligne octale (`\012`) et que vous essayez de charger ces données avec le paramètre ESCAPE, Amazon Redshift charge la valeur `012` dans la table et ne l'interprète pas comme un saut de ligne qui est en cours d'échappement.

Pour insérer des caractère de saut ligne dans les données qui proviennent de plateformes Microsoft Windows, vous devrez peut-être utiliser deux caractères d'échappement : un pour le retour chariot et un autre pour le saut de ligne. Sinon, vous pouvez supprimer les retours chariot avant de charger le fichier (par exemple, en utilisant l'utilitaire dos2unix).

EXPLICIT_IDS

Utilisez EXPLICIT_IDS avec des tables ayant des colonnes IDENTITY si vous souhaitez remplacer les valeurs générées automatiquement par des valeurs explicites dans les fichiers de données source pour les tables. Si la commande inclut une liste de colonnes, cette liste doit inclure les colonnes IDENTITY pour utiliser ce paramètre. Le format de données pour les valeurs EXPLICIT_IDS doit correspondre au format IDENTITY spécifié par la définition CREATE TABLE.

Lorsque vous exécutez une commande COPY sur une table avec l'option EXPLICIT_IDS, Amazon Redshift ne vérifie plus l'unicité des colonnes IDENTITY dans la table.

Si une colonne est définie avec GENERATED BY DEFAULT AS IDENTITY, elle peut être copiée. Les valeurs sont générées ou mises à jour avec des valeurs que vous fournissez. L'option EXPLICIT_IDS n'est pas obligatoire. COPY ne met pas à jour le filigrane élevé d'identité.

Pour un exemple de commande COPY utilisant EXPLICIT_IDS, consultez Charger la table VENUE avec des valeurs EXPLICIT pour une colonne IDENTITY.

FILLRECORD

Autorise le chargement des fichiers de données lorsqu'il manque des colonnes contiguës à la fin de certains des enregistrements. Les colonnes manquantes sont chargées en tant que NULLs.

Pour les formats texte et CSV, si la colonne manquante est une colonne VARCHAR, des chaînes de longueur nulle sont chargées à la place de. NULLs Pour charger des colonnes NULLs dans VARCHAR à partir de texte et de CSV, spécifiez le mot clé EMPTYASNULL. La substitution NULL ne fonctionne que si la définition de colonne le permet NULLs.

Par exemple, si la définition de table contient quatre colonnes CHAR qui autorisent la valeur Null, et qu'un enregistrement contient les valeurs `apple, orange, banana, mango`, la commande COPY peut charger et remplir un enregistrement qui contient uniquement les valeurs `apple, orange`. Les valeurs CHAR manquantes seraient chargées en tant que valeurs NULL.

IGNOREBLANKLINES

Ignore les lignes vides qui contiennent uniquement un saut de ligne dans un fichier de données et n'essaie pas de les charger.

IGNOREHEADER [AS] number_rows

Traite les number_rows spécifiées comme un en-tête de fichier et ne les charge pas. Utilisez IGNOREHEADER pour ignorer les en-têtes de fichier dans tous les fichiers d'une charge parallèle.

NULL AS 'null_string'

Charge les champs qui mettent en correspondance les null_string comme NULL, où null_string peut être une chaîne. Si vos données incluent une marque de fin null, également appelée NUL (UTF-8 0000) ou zéro binaire (0x000), la commande COPY la traite comme tout autre caractère. Par exemple, un enregistrement contenant '1' || NUL || '2' est copié sous la forme d'une chaîne d'une longueur de 3 octets. Si un champ contient uniquement NUL, vous pouvez utiliser NULL AS pour remplacer la marque de fin null par NULL en spécifiant `'\0'` ou `'\000'`, par exemple, `NULL AS '\0'` ou `NULL AS '\000'`. Si un champ qui contient une chaîne qui se termine par NUL et NULL AS est spécifié, la chaîne est insérée avec NUL à la fin. N'utilisez pas '\n' (saut de ligne) pour la valeur null_string. Amazon Redshift réserve '\n' pour l'utiliser en tant que délimiteur de ligne. La valeur par défaut null_string est `'\N`'.

> **Note**
>
> Si vous essayez de charger les valeurs null dans une colonne définie comme NOT NULL, la commande COPY échoue.

REMOVEQUOTES

Supprime les guillemets des chaînes dans les données entrantes. Tous les caractères compris entre les guillemets, y compris les délimiteurs, sont conservés. Si une chaîne commence par un guillemet anglais ou des guillemets doubles, sans caractère de fin correspondant, la commande COPY échoue à charger cette ligne et renvoie une erreur. Le tableau suivant présente quelques exemples simples de chaînes contenant des guillemets et les valeurs chargées en conséquence.

Chaîne d'entrée	Valeur chargée avec l'option REMOVEQUOTES
"The delimiter is a pipe (|) character"	The delimiter is a pipe (|) character
'Black'	Black
"White"	White
Blue'	Blue'
Blue'	Valeur non chargée : condition d'erreur
"Blue	Valeur non chargée : condition d'erreur
' ' 'Black' ' '	' 'Black' '
' '	<white space>

ROUNDEC

Arrondit les valeurs numériques lorsque l'échelle de la valeur d'entrée est supérieure à l'échelle de la colonne. Par défaut, la commande COPY tronque les valeurs lorsque cela est nécessaire pour s'adapter à l'échelle de la colonne. Par exemple, si une valeur de `20.259` est chargée dans une colonne DECIMAL(8,2), la commande COPY tronque la valeur de `20.25` par défaut. Si ROUNDEC est spécifié, la commande COPY arrondit la valeur de `20.26`. La commande INSERT arrondit toujours les valeurs autant que possible afin de les adapter à l'échelle de la colonne. Une commande COPY avec le paramètre ROUNDEC se comporte donc de la même manière qu'une commande INSERT.

TIMEFORMAT [AS] {'timeformat_string' | 'auto' | 'epochsecs' | 'epochmillisecs' }

Spécifie le format de l'heure. Si aucun TIMEFORMAT n'est spécifié, le format par défaut est `YYYY-MM-DD HH:MI:SS` pour les colonnes TIMESTAMP ou `YYYY-MM-DD HH:MI:SSOF` pour les colonnes TIMESTAMPTZ, `OF` correspondant au décalage par rapport à l'heure UTC (Coordinated Universal Time). Vous ne pouvez pas inclure de spécificateur de fuseau horaire dans timeformat_string. Pour charger les données TIMESTAMPTZ dont le format est différent du format par défaut, spécifiez « auto ». Pour plus d'informations, consultez Utilisation de la reconnaissance automatique avec DATEFORMAT et TIMEFORMAT. Pour plus d'informations sur timeformat_string, consultez Chaînes DATEFORMAT et TIMEFORMAT.

L'argument `'auto'` reconnaît plusieurs formats qui ne sont pas pris en charge lors de l'utilisation d'une chaîne DATEFORMAT et TIMEFORMAT. Si la commande COPY ne reconnaît pas le format de vos valeurs de date ou d'heure, ou si vos valeurs de date et d'heure utilisent des formats différents les uns des autres, utilisez l'argument `'auto'` avec le paramètre DATEFORMAT ou TIMEFORMAT. Pour plus d'informations, consultez Utilisation de la reconnaissance automatique avec DATEFORMAT et TIMEFORMAT.

Si vos données source sont représentées sous la forme de l'heure d'époque, qui est le nombre de secondes ou de millisecondes depuis le 1er janvier 1970, 00:00:00 UTC, précisez `'epochsecs'` ou `'epochmillisecs'`.

Les mots-clés `'auto'`, `'epochsecs'`, et `'epochmillisecs'` sont sensibles à la casse.

Le mot-clé AS est facultatif.

TRIMBLANKS

Supprime les caractères d'espace vide de fin d'une chaîne VARCHAR. Ce paramètre s'applique uniquement aux colonnes avec un type de données VARCHAR.

TRUNCATECOLUMNS

Tronque les données des colonnes avec le nombre de caractères donné afin qu'il corresponde à la spécification de la colonne. S'applique uniquement aux colonnes avec un type de données VARCHAR ou CHAR et des lignes de 4 Mo ou moins.

Opérations de chargement de données

Gérez le comportement par défaut de l'opération de chargement pour le dépannage ou pour réduire les temps de chargement en spécifiant les paramètres suivants.

- COMPROWS
- COMPUPDATE
- IGNOREALLERRORS
- MAXERROR
- NOLOAD
- STATUPDATE

Paramètres

COMPROWS numrows

Permet de spécifier le nombre de lignes à utiliser comme taille d'échantillon pour l'analyse de la compression. L'analyse est exécutée sur les lignes de chaque tranche de données. Par exemple, si vous spécifiez `COMPROWS 1000000` (1 000 000) et que le système contient quatre sections totales, pas plus de 250 000 lignes par section sont lues et analysées.

Si le paramètre COMPROWS n'est pas spécifié, la taille de l'échantillon par défaut est de 100 000 par tranche. Les valeurs du paramètre COMPROWS inférieures à la valeur par défaut de 100 000 lignes par tranche sont automatiquement mises à niveau vers la valeur par défaut. Toutefois, la compression automatique n'aura pas lieu si la quantité de données en cours de chargement n'est pas suffisante pour produire un échantillon représentatif.

Si le nombre défini pour le paramètre COMPROWS dépasse le nombre de lignes dans le fichier d'entrée, la commande COPY produit et exécute toujours l'analyse de la compression sur toutes les lignes disponibles. La plage acceptée pour cet argument est un nombre compris entre 1000 et 2147483647 (2 147 483 647).

COMPUPDATE [PRESET | { ON | TRUE } | { OFF | FALSE }],

Permet de contrôler si les encodages de compression sont appliqués automatiquement pendant une exécution de la commande COPY.

Lorsque COMPUPDATE utilise PRESET, la commande COPY choisit l'encodage de compression pour chaque colonne si la table de la cible est vide, même si les colonnes ont déjà des encodages autres que RAW. Les encodages de colonne actuellement spécifiés peuvent être remplacés. L'encodage de chaque colonne est basé sur le type de données de la colonne. Aucune donnée échantillonnée. Amazon Redshift attribue automatiquement l'encodage de compression comme suit :

- Les colonnes qui sont définies comme des clés de tri se voient attribuer une compression RAW.
- Les colonnes qui sont définies comme des types de données BOOLEAN, REAL ou DOUBLE PRECISION se voient attribuer une compression RAW.
- Les colonnes définies comme SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIMESTAMP ou TIMESTAMPTZ sont compressées. AZ64
- Les colonnes définies comme CHAR ou VARCHAR sont affectées à la compression LZO.

Lorsque COMPUPDATE est omis, la commande COPY choisit l'encodage de compression pour chaque colonne uniquement si la table de la cible est vide et si vous n'avez pas spécifié d'encodage (autre que BRUT) pour les colonnes. L'encodage pour chaque colonne est déterminé par Amazon Redshift. Aucune donnée échantillonnée.

Lorsque le paramètre COMPUPDATE est défini sur ON (ou TRUE) ou que le paramètre COMPUPDATE est spécifié sans une option, la commande COPY applique la compression automatique si la table est vide, même si les colonnes de la table contiennent déjà des encodages autres que RAW. Les encodages de colonne actuellement spécifiés peuvent être remplacés. L'encodage de chaque colonne est basé sur une analyse des exemples de données. Pour plus d'informations, consultez Chargement des tables avec compression automatique.

Lorsque le paramètre COMPUPDATE est défini sur OFF (ou FALSE), la compression automatique est désactivée. Les encodages de colonne ne sont pas modifiés.

Pour plus de détails sur la table système utilisée pour analyser la compression, consultez STL_ANALYZE_COMPRESSION.

IGNOREALLERRORS

Vous pouvez spécifier cette option pour ignorer toutes les erreurs qui se produisent pendant l'opération de chargement.

Vous ne pouvez pas spécifier l'option IGNOREALLERRORS si vous spécifiez l'option MAXERROR. Vous ne pouvez pas spécifier l'option IGNOREALLERRORS pour les formats de données en colonnes tels que ORC et Parquet.

MAXERROR [AS] error_count

Si la charge renvoie le nombre d'erreurs error_count ou un nombre supérieur, la charge échoue. Si la charge renvoie moins d'erreurs, elle continue et renvoie un message INFO qui indique le nombre de lignes qui n'a pas pu être chargé. Utilisez ce paramètre pour permettre aux charges de continuer lorsque certaines lignes échouent à se charger dans la table en raison d'erreurs de mise en forme ou d'autres incohérences dans les données.

Définissez cette valeur sur `0` ou `1` si vous voulez que la charge échoue dès que la première erreur se produit. Le mot-clé AS est facultatif. La valeur par défaut MAXERROR est `0` et la limite est `100000`.

Le nombre d'erreurs réel signalé peut être supérieur à la valeur du paramètre MAXERROR spécifié en raison de la nature parallèle d'Amazon Redshift. Si un nœud dans le cluster Amazon Redshift détecte que la valeur du paramètre MAXERROR a été dépassée, chaque nœud indique toutes les erreurs qu'il a rencontrées.

NOLOAD

Permet de vérifier la validité du fichier de données sans réellement charger les données. Utilisez le paramètre NOLOAD pour vous assurer que votre fichier de données se charge sans erreur avant d'exécuter la charge de données réelle. L'exécution de COPY avec le paramètre NOLOAD est beaucoup plus rapide que le chargement des données, car il analyse uniquement les fichiers.

STATUPDATE [{ ON | TRUE } | { OFF | FALSE }]

Permet de définir le calcul automatique et l'actualisation des statistiques de l'optimiseur à la fin d'une commande COPY réussie. Par défaut, si le paramètre STATUPDATE n'est pas utilisé, les statistiques sont mises à jour automatiquement si la table est initialement vide.

Chaque fois qu'une intégration de données dans une table non vide modifie considérablement la taille de la table, nous recommandons la mise à jour des statistiques en exécutant une commande ANALYSE ou en utilisant l'argument STATUPDATE ON.

Avec le paramètre STATUPDATE ON (ou TRUE), les statistiques sont mises à jour automatiquement, que la table soit initialement vide ou non. Si le paramètre STATUPDATE est utilisé, l'utilisateur actuel doit être soit propriétaire de la table soit un super-utilisateur. Si le paramètre STATUPDATE n'est pas spécifié, seule l'autorisation INSERT est obligatoire.

Avec le paramètre STATUPDATE OFF (ou FALSE), les statistiques ne sont jamais mises à jour.

Pour plus d'informations, consultez Analyse des tables.

Liste alphabétique des paramètres

La liste suivante fournit des liens vers chaque description du paramètre de commande COPY, par ordre alphabétique.

- ACCEPTANYDATE

- ACCEPTINVCHARS
- the section called “ID DE CLÉ D'ACCÈS, CLÉ D'ACCÈS SECRÈTE”
- AVRO
- BLANKSASNULL
- BZIP2
- COMPROWS
- COMPUPDATE
- the section called “CREDENTIALS”
- CSV
- DATEFORMAT
- DELIMITER
- EMPTYASNULL
- ENCODING
- ENCRYPTED
- ESCAPE
- EXPLICIT_IDS
- FILLRECORD
- FIXEDWIDTH
- FORMAT
- FROM
- GZIP
- the section called “IAM_ROLE”
- IGNOREALLERRORS
- IGNOREBLANKLINES
- IGNOREHEADER
- JSON
- LZOP
- MANIFEST
- MASTER_SYMMETRIC_KEY

- MAXERROR
- NOLOAD
- NULL AS
- READRATIO
- REGION
- REMOVEQUOTES
- ROUNDEC
- the section called “SESSION_TOKEN”
- SHAPEFILE
- SSH
- STATUPDATE
- TIMEFORMAT
- TRIMBLANKS
- TRUNCATECOLUMNS
- ZSTD

Notes d’utilisation

Rubriques

- Autorisations d’accès aux autres ressources AWS
- Utilisation de COPY avec des alias de point d’accès Amazon S3
- Chargement de données multioctets à partir d'Amazon S3
- Chargement d’une colonne avec le type de données GEOMETRY ou GEOGRAPHY
- Chargement du type de données HLLSKETCH
- Chargement d’une colonne avec le type de données VARBYTE
- Erreurs survenant lors de la lecture de plusieurs fichiers
- Exécution de la commande COPY à partir du format JSON
- COPY depuis les formats de données en colonnes
- Chaînes DATEFORMAT et TIMEFORMAT
- Utilisation de la reconnaissance automatique avec DATEFORMAT et TIMEFORMAT

Autorisations d’accès aux autres ressources AWS

Pour déplacer des données entre votre cluster et une autre AWS ressource, telle qu'Amazon S3, Amazon DynamoDB, Amazon EMR ou EC2 Amazon, votre cluster doit être autorisé à accéder à la ressource et à effectuer les actions nécessaires. Par exemple, pour charger des données depuis Amazon S3, la commande COPY doit disposer de l’accès LIST au compartiment et de l’accès GET pour les objets du compartiment. Pour plus d’informations sur les autorisations minimales, consultez Autorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY.

Pour obtenir l’autorisation d’accéder à la ressource, votre cluster doit être authentifié. Vous pouvez choisir l’une des méthodes d’authentification suivantes :

- Contrôle d’accès basé sur les rôles— Pour le contrôle d'accès basé sur les rôles, vous spécifiez un rôle AWS Identity and Access Management (IAM) que votre cluster utilise pour l'authentification et l'autorisation. Pour protéger vos AWS informations d'identification et vos données sensibles, nous vous recommandons vivement d'utiliser l'authentification basée sur les rôles.
- Contrôle d’accès basé sur les clés— Pour le contrôle d'accès par clé, vous fournissez les informations d' AWS accès (ID de clé d'accès et clé d'accès secrète) d'un utilisateur sous forme de texte brut.

Contrôle d’accès basé sur les rôles

Avec un contrôle d’accès basé sur les rôles, votre cluster assume temporairement un rôle IAM en votre nom. Puis, selon les autorisations accordées au rôle, votre cluster peut accéder aux ressources AWS requises.

La création d’un rôle IAM est similaire à l’octroi d’autorisations à un utilisateur, car il s’agit d’une identité AWS avec des politiques d’autorisation qui déterminent ce que l’identité peut et ne peut pas faire dans AWS. En revanche, plutôt que d’être associé de manière unique à un utilisateur, un rôle peut être assumé par toute entité le nécessitant. En outre, aucune information d’identification (mot de passe ou clés d’accès) n’est associée à celui-ci. Au lieu de cela, si un rôle est associé à un cluster, les clés d’accès sont créées de manière dynamique et fournies au cluster.

Nous vous recommandons d'utiliser le contrôle d'accès basé sur les rôles, car il permet un contrôle plus sûr et plus précis de l'accès aux AWS ressources et aux données utilisateur sensibles, en plus de protéger vos informations d'identification. AWS

L’authentification basée sur les rôles offre les avantages suivants :

- Vous pouvez utiliser les outils IAM AWS standard pour définir un rôle IAM et l'associer à plusieurs clusters. Lorsque vous modifiez la stratégie d’accès d’un rôle, les modifications sont automatiquement appliquées à tous les clusters qui utilisent ce rôle.
- Vous pouvez définir des politiques IAM détaillées qui accordent des autorisations à des clusters et à des utilisateurs de base de données spécifiques pour accéder à des AWS ressources et à des actions spécifiques.
- Votre cluster obtient les informations d’identification de séance temporaires au moment de l’exécution et les actualise en fonction des besoins jusqu’à la fin de l’opération. Si vous utilisez les informations d’identification temporaires basées sur la clé, l’opération échoue si celles-ci expirent avant la fin.
- Votre ID de clé d’accès et ID de clé d’accès secrète ne sont pas stockés ou transmis dans votre code SQL.

Pour utiliser le contrôle d’accès basé sur les rôles, vous devez d’abord créer un rôle IAM à l’aide du type de rôle de service Amazon Redshift, puis attacher le rôle à votre cluster. Le rôle doit avoir, au minimum, les autorisations répertoriées dans Autorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY. Pour savoir comment créer un rôle IAM et l'associer à votre cluster, consultez la section Autoriser Amazon Redshift à accéder à AWS d'autres services en votre nom dans le guide de gestion Amazon Redshift.

Vous pouvez ajouter un rôle à un cluster ou afficher les rôles associés à un cluster à l’aide de la console de gestion, de la CLI ou de l’API Amazon Redshift. Pour plus d’informations, consultez Associer un rôle IAM à un cluster dans le Guide de gestion Amazon Redshift.

Lorsque vous créez un rôle IAM, IAM renvoie un Amazon Resource Name (ARN) pour le rôle. Pour spécifier un rôle IAM, indiquez l’ARN de rôle avec le paramètre the section called “Utilisation de IAM_ROLE” ou le paramètre the section called “Utilisation des informations d'identification”.

Par exemple, supposons que le rôle suivant est attaché au cluster.

```
"IamRoleArn": "arn:aws:iam::0123456789012:role/MyRedshiftRole"
```

L’exemple suivant de commande COPY utilise le paramètre IAM_ROLE avec l’ARN dans l’exemple précédent pour l’authentification et l’accès à Amazon S3.

```
copy customer from 's3://amzn-s3-demo-bucket/mydata'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

L'exemple suivant de commande COPY utilise le paramètre CREDENTIALS pour spécifier le rôle IAM.

```
copy customer from 's3://amzn-s3-demo-bucket/mydata'
credentials
'aws_iam_role=arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

En outre, un super-utilisateur peut accorder le privilège ASSUMEROLE à des utilisateurs et groupes de base de données afin de fournir l'accès à un rôle pour les opérations COPY. Pour plus d'informations, consultez GRANT.

Contrôle d'accès basé sur les clés

Avec le contrôle d'accès basé sur les clés, vous fournissez l'ID de clé d'accès et la clé d'accès secrète à un utilisateur IAM autorisé à accéder aux AWS ressources contenant les données. Vous pouvez utiliser les the section called "Utilisation de ACCESS_KEY_ID et SECRET_ACCESS_KEY" paramètres ensemble ou le the section called "Utilisation des informations d'identification" paramètre.

> **Note**
>
> Nous vous recommandons vivement d'utiliser un rôle IAM pour l'authentification au lieu de fournir une clé d'accès secrète et un ID de clé d'accès en texte brut. Si vous optez pour le contrôle d'accès par clé, n'utilisez jamais les informations d'identification de votre AWS compte (root). Créez toujours un utilisateur IAM et fournissez l'ID de clé d'accès et la clé d'accès secrète de cet utilisateur. Pour connaître les étapes à suivre pour créer un utilisateur IAM, consultez Création d'un utilisateur IAM dans votre compte AWS.

Pour vous authentifier à l'aide de ACCESS_KEY_ID et SECRET_ACCESS_KEY, remplacez et par l'ID de clé d'accès *<access-key-id>* et la clé *<secret-access-key>* d'accès secrète complète d'un utilisateur autorisé, comme indiqué ci-dessous.

```
ACCESS_KEY_ID '<access-key-id>'
SECRET_ACCESS_KEY '<secret-access-key>';
```

Pour vous authentifier à l'aide du paramètre CREDENTIALS, remplacez *<access-key-id>* et *<secret-access-key>* par l'ID de clé d'accès d'un utilisateur autorisé et la clé d'accès secrète complète, comme indiqué ci-dessous.

```
CREDENTIALS
'aws_access_key_id=<access-key-id>;aws_secret_access_key=<secret-access-key>';
```

L'utilisateur IAM doit avoir, au minimum, les autorisations répertoriées dans Autorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY.

informations d'identification de sécurité temporaires

Si vous utilisez un contrôle d'accès basé sur la clé, vous pouvez limiter davantage l'accès dont disposent les utilisateurs à vos données en utilisant des informations d'identification de sécurité temporaires. L'authentification basée sur les rôles utilise automatiquement des informations d'identification temporaires.

> **Note**
>
> Nous vous recommandons vivement d'utiliser role-based access control plutôt que de créer des identifiants temporaires et de fournir un ID de clé d'accès et une clé d'accès secrète sous forme de texte brut. Le contrôle d'accès basé sur les rôles utilise automatiquement des informations d'identification temporaires.

Les informations d'identification de sécurité temporaires offrent une sécurité améliorée parce qu'elles sont de courte durée et ne peuvent pas être réutilisées après leur expiration. L'ID de clé d'accès et la clé d'accès secrète générés avec le jeton ne peuvent pas être utilisés sans le jeton et un utilisateur qui dispose de ces informations d'identification de sécurité temporaires peut accéder à vos ressources uniquement jusqu'à ce que les informations d'identification expirent.

Pour accorder aux utilisateurs un accès temporaire à vos ressources, vous devez appeler les opérations d'API AWS Security Token Service (AWS STS). Les opérations AWS STS d'API renvoient des informations d'identification de sécurité temporaires composées d'un jeton de sécurité, d'un identifiant de clé d'accès et d'une clé d'accès secrète. Vous pouvez délivrer des informations d'identification de sécurité temporaires aux utilisateurs qui doivent accéder temporairement à vos ressources. Ces utilisateurs peuvent être des utilisateurs IAM existants, ou bien ils peuvent être des utilisateurs non-AWS . Pour plus d'informations sur les informations d'identification de sécurité temporaires, consultez Informations d'identification de sécurité temporaires dans le Guide de l'utilisateur IAM.

Vous pouvez utiliser les paramètres the section called "Utilisation de ACCESS_KEY_ID et SECRET_ACCESS_KEY" ensemble avec le paramètre the section called "SESSION_TOKEN" ou

le paramètre the section called “Utilisation des informations d'identification”. Vous devez également fournir l’ID de clé d’accès et la clé d’accès secrète qui ont été fournis avec le jeton.

Pour vous authentifier à l'aide de ACCESS_KEY_ID, SECRET_ACCESS_KEY et SESSION_TOKEN, remplacez et comme indiqué ci-dessous. *<temporary-access-key-id>* *<temporary-secret-access-key>* *<temporary-token>*

```
ACCESS_KEY_ID '<temporary-access-key-id>'
SECRET_ACCESS_KEY '<temporary-secret-access-key>'
SESSION_TOKEN '<temporary-token>';
```

Pour vous authentifier à l’aide de CREDENTIALS, ajoutez `session_token=`*`<temporary-token>`* dans la chaîne d’informations d’identification, comme indiqué ci-après.

```
CREDENTIALS
'aws_access_key_id=<temporary-access-key-id>;aws_secret_access_key=<temporary-secret-access-key>;session_token=<temporary-token>';
```

Une commande COPY avec des informations d’identification de sécurité temporaires est indiquée dans l’exemple suivant.

```
copy table-name
from 's3://objectpath'
access_key_id '<temporary-access-key-id>'
secret_access_key '<temporary-secret-access-key>'
session_token '<temporary-token>';
```

L’exemple suivant charge la table LISTING à l’aide des informations d’identification temporaires et de chiffrement de fichier.

```
copy listing
from 's3://amzn-s3-demo-bucket/data/listings_pipe.txt'
access_key_id '<temporary-access-key-id>'
secret_access_key '<temporary-secret-access-key>'
session_token '<temporary-token>'
master_symmetric_key '<root-key>'
encrypted;
```

L’exemple suivant charge la table LISTING à l’aide du paramètre CREDENTIALS avec des informations d’identification temporaires et de chiffrement de fichier.

```
copy listing
from 's3://amzn-s3-demo-bucket/data/listings_pipe.txt'
credentials
'aws_access_key_id=<temporary-access-key-id>;aws_secret_access_key=<temporary-secret-access-key>;session_token=<temporary-token>;master_symmetric_key=<root-key>'
encrypted;
```

Important

Les informations d'identification de sécurité temporaires doivent être valides pour toute la durée de l'opération COPY ou UNLOAD. Si les informations d'identification de sécurité temporaires arrivent à expiration au cours de l'opération, la commande échoue et la transaction est annulée. Par exemple, si les informations d'identification de sécurité temporaires expirent au bout de 15 minutes et que l'opération COPY dure une heure, cette dernière échouera avant d'être terminée. Si vous utilisez l'accès basé sur les rôles, les informations d'identification de sécurité temporaires sont automatiquement actualisées jusqu'à la fin de l'opération.

Autorisations IAM pour les commandes COPY, UNLOAD et CREATE LIBRARY

Le rôle IAM ou l'utilisateur référencé par le paramètre CREDENTIALS doit disposer, au minimum, des autorisations suivantes :

- Pour exécuter la commande COPY depuis Amazon S3, l'autorisation d'exécuter LIST sur le compartiment Amazon S3 et d'exécuter GET sur les objets Amazon S3 en cours de chargement, ainsi que le fichier manifeste, le cas échéant.
- Pour COPIER depuis Amazon S3, Amazon EMR et des hôtes distants (SSH) avec des données au format JSON, autorisation de LISTER et d'OBTENIR le fichier sur JSONPaths Amazon S3, le cas échéant.
- Pour exécuter la commande COPY depuis DynamoDB, l'autorisation d'exécuter SCAN et DESCRIBE sur la table DynamoDB qui est en cours de chargement.
- Pour exécuter la commande COPY depuis un cluster Amazon EMR, l'autorisation pour l'action `ListInstances` sur le cluster Amazon EMR.
- Pour exécuter la commande UNLOAD sur Amazon S3, les autorisations d'exécuter GET, LIST et PUT pour le compartiment Amazon S3 vers lequel les fichiers de données sont en cours de déchargement.

- Pour exécuter la commande CREATE LIBRARY depuis Amazon S3, l'autorisation d'exécuter LIST sur le compartiment Amazon S3 et d'exécuter GET sur les objets Amazon S3 qui sont importés

> **Note**
>
> Si vous recevez le message d'erreur `S3ServiceException: Access Denied`, lorsque vous exécutez une commande COPY, UNLOAD ou CREATE LIBRARY, votre cluster ne dispose pas des autorisations d'accès appropriées pour Amazon S3.

Vous pouvez gérer les autorisations IAM en attachant une politique IAM à un rôle IAM attaché à votre cluster, à votre utilisateur ou au groupe auquel appartient votre utilisateur. Par exemple, la politique gérée par `AmazonS3ReadOnlyAccess` accorde les autorisations LIST et GET aux ressources Amazon S3. Pour plus d'informations sur les politiques IAM, consultez Gestion des politiques IAM dans le Guide de l'utilisateur IAM.

Utilisation de COPY avec des alias de point d'accès Amazon S3

COPY prend en charge les alias de point d'accès Amazon S3. Pour plus d'informations, consultez Utilisation d'un alias de type compartiment pour votre point d'accès dans le Guide de l'utilisateur Amazon Simple Storage Service.

Chargement de données multioctets à partir d'Amazon S3

Si vos données incluent des caractères non-ASCII codés sur plusieurs octets (par exemple, les caractères chinois ou cyrilliques), vous devez charger les données dans des colonnes VARCHAR. Le type de données VARCHAR prend en charge les caractères UTF-8 codés sur quatre octets, mais le type de données CHAR n'accepte que les caractères ASCII codés sur un octet. Vous ne pouvez pas charger de caractères codés sur cinq octets ou plus dans des tables Amazon Redshift. Pour plus d'informations, consultez Caractères multioctets.

Chargement d'une colonne avec le type de données GEOMETRY ou GEOGRAPHY

Vous ne pouvez exécuter la commande COPY vers des colonnes `GEOMETRY` ou `GEOGRAPHY` qu'à partir de données d'un fichier texte séparé par des caractères, tel qu'un fichier CSV. Les données doivent être sous la forme hexadécimale du format well-known binary (WKB ou EWKB) ou du format well-known text (WKT ou EWKT) et correspondre à la taille maximale d'une ligne d'entrée unique à la commande COPY. Pour plus d'informations, consultez COPY.

Pour plus d'informations sur la façon de charger à partir d'un fichier de formes, consultez Chargement d'un shapefile dans Amazon Redshift.

Pour plus d'informations sur le type de données GEOMETRY ou GEOGRAPHY, consultez Interrogation des données spatiales dans Amazon Redshift.

Chargement du type de données HLLSKETCH

Vous pouvez copier les esquisses HLL uniquement dans un format fragmenté ou dense pris en charge par Amazon Redshift. Pour utiliser la commande COPY sur HyperLogLog des esquisses, utilisez le format Base64 pour les HyperLogLog esquisses denses et le format JSON pour les esquisses clairsemées. HyperLogLog Pour de plus amples informations, veuillez consulter HyperLogLog fonctions.

L'exemple suivant importe les données d'un fichier CSV dans une table à l'aide des commandes CREATE TABLE et COPY. Tout d'abord, l'exemple crée la table t1 à l'aide de la commande CREATE TABLE.

```
CREATE TABLE t1 (sketch hllsketch, a bigint);
```

Ensuite, il utilise la commande COPY pour importer des données d'un fichier CSV dans la table t1.

```
COPY t1 FROM s3://amzn-s3-demo-bucket/unload/' IAM_ROLE
 'arn:aws:iam::0123456789012:role/MyRedshiftRole' NULL AS 'null' CSV;
```

Chargement d'une colonne avec le type de données VARBYTE

Vous pouvez charger des données à partir d'un fichier au format CSV, Parquet et ORC. Pour CSV, les données sont chargées à partir d'un fichier en représentation hexadécimale des données VARBYTE. Vous ne pouvez pas charger les données VARBYTE avec l'option FIXEDWIDTH. L'option ADDQUOTES ou REMOVEQUOTES de COPY n'est pas prise en charge. Une colonne VARBYTE ne peut pas être utilisée comme colonne de partition.

Erreurs survenant lors de la lecture de plusieurs fichiers

La commande COPY est atomique et transactionnelle. En d'autres termes, même lorsque la commande COPY lit des données de plusieurs fichiers, l'ensemble du processus est considéré comme une seule opération. Si la commande COPY rencontre une erreur de lecture d'un fichier, elle refait automatiquement des tentatives jusqu'à ce que le processus expire (voir statement_timeout) ou si les données ne peuvent pas être téléchargées depuis Amazon S3 pendant une période prolongée

(entre 15 et 30 minutes), en s'assurant que chaque fichier est chargé une seule fois. En cas d'échec de la commande COPY, l'ensemble de la transaction est annulé et toutes les modifications sont annulées. Pour plus d'informations sur la gestion des erreurs de chargement, consultez Résolution des problèmes de chargement de données.

Une fois qu'une commande COPY est lancée avec succès, elle n'échoue pas si la séance s'arrête, par exemple lorsque le client se déconnecte. Toutefois, si la commande COPY se trouve dans bloc de transaction BEGIN … END qui ne se termine pas, car la séance s'arrête, toute la transaction, y compris la commande COPY, est annulée. Pour plus d'informations sur les transactions, consultez BEGIN.

Exécution de la commande COPY à partir du format JSON

La structure des données JSON se compose d'un ensemble d'objets ou de tableaux. Un objet JSON commence et finit par des accolades, et contient une collection non ordonnée de paires nom-valeur. Chaque paire de nom et de valeur est séparée par deux points, et les paires sont séparées par des virgules. Le nom est une chaîne entre guillemets doubles. Les guillemets doivent être des guillemets simples (0x22), ni culbutés, ni courbes.

Un tableau JSON commence et finit par des crochets, et contient une collection ordonnée de valeurs séparées par des virgules. Une valeur peut être une chaîne comprise entre des guillemets doubles, un nombre, une valeur booléenne true ou false, null, un objet JSON ou tableau.

Les tableaux et objets JSON peuvent être imbriqués, ce qui permet d'obtenir une structure de données hiérarchique. L'exemple suivant illustre une structure de données JSON avec deux objets valides.

```
{
    "id": 1006410,
    "title": "Amazon Redshift Database Developer Guide"
}
{
    "id": 100540,
    "name": "Amazon Simple Storage Service User Guide"
}
```

L'exemple suivant illustre les mêmes données sous forme de deux tableaux JSON.

```
[
    1006410,
    "Amazon Redshift Database Developer Guide"
```

```
]
[
    100540,
    "Amazon Simple Storage Service User Guide"
]
```

Options COPY pour JSON

Vous pouvez spécifier les options suivantes lorsque vous utilisez la commande COPY avec des données au format JSON :

- `'auto'` – La commande COPY charge automatiquement les champs à partir du fichier JSON.
- `'auto ignorecase'` – La commande COPY charge automatiquement les champs à partir du fichier JSON tout en ignorant la casse des noms de champs.
- `s3://jsonpaths_file`— COPY utilise un JSONPaths fichier pour analyser les données source JSON. Un JSONPaths fichier est un fichier texte qui contient un seul objet JSON dont le nom est `"jsonpaths"` associé à un tableau d' JSONPath expressions. Si le nom est une chaîne autre que`"jsonpaths"`, COPY utilise l'`'auto'`argument au lieu du JSONPaths fichier.

Pour des exemples montrant comment charger des données à l'aide de `'auto'``'auto ignorecase'`, ou d'un JSONPaths fichier, et à l'aide d'objets ou de tableaux JSON, consultezCopier à partir d’exemples JSON.

JSONPath option

Dans la syntaxe COPY d'Amazon Redshift, une JSONPath expression indique le chemin explicite vers un élément de nom unique dans une structure de données hiérarchique JSON, en utilisant soit la notation entre crochets, soit la notation par points. Amazon Redshift ne prend en charge aucun JSONPath élément, tel que les caractères génériques ou les expressions de filtre, susceptibles de donner lieu à un chemin ambigu ou à plusieurs éléments de nom. Par conséquent, Amazon Redshift ne peut pas analyser des structures de données complexes, à plusieurs niveaux.

Voici un exemple de JSONPaths fichier contenant des JSONPath expressions utilisant la notation entre crochets. Le symbole du dollar ($) représente la structure de niveau racine.

```
{
    "jsonpaths": [
       "$['id']",
       "$['store']['book']['title']",
 "$['location'][0]"
```

```
    ]
}
```

Dans l'exemple précédent, `$['location'][0]` fait référence au premier élément d'un tableau. JSON utilise l'indexation de tableau de base zéro. Les index du tableau doivent être des nombres entiers positifs (supérieurs ou égaux à zéro).

L'exemple suivant montre le JSONPaths fichier précédent en utilisant la notation par points.

```
{
    "jsonpaths": [
       "$.id",
       "$.store.book.title",
 "$.location[0]"
    ]
}
```

Vous ne pouvez pas combiner la notation d'accolades et la notation de points dans le tableau `jsonpaths`. Les accolades peuvent être utilisées dans la notation d'accolades et la notation de points pour faire référence à un élément du tableau.

Lorsque vous utilisez la notation par points, les JSONPath expressions ne peuvent pas contenir les caractères suivants :

- Guillemet anglais (')
- Point (.)
- Crochets ([ ]), sauf pour faire référence à un élément de tableau

Si la valeur de la paire nom-valeur référencée par une JSONPath expression est un objet ou un tableau, l'objet ou le tableau entier est chargé sous forme de chaîne, y compris les accolades. Par exemple, supposons que vos données JSON contiennent l'objet suivant.

```
{
    "id": 0,
    "guid": "84512477-fa49-456b-b407-581d0d851c3c",
    "isActive": true,
    "tags": [
        "nisi",
        "culpa",
        "ad",
```

```
        "amet",
        "voluptate",
        "reprehenderit",
        "veniam"
    ],
    "friends": [
        {
            "id": 0,
            "name": "Martha Rivera"
        },
        {
            "id": 1,
            "name": "Renaldo"
        }
    ]
}
```

L' JSONPath expression renvoie `$['tags']` ensuite la valeur suivante.

```
"["nisi","culpa","ad","amet","voluptate","reprehenderit","veniam"]"
```

L' JSONPath expression renvoie `$['friends'][1]` ensuite la valeur suivante.

```
"{"id": 1,"name": "Renaldo"}"
```

Chaque JSONPath expression du `jsonpaths` tableau correspond à une colonne de la table cible Amazon Redshift. L'ordre des éléments du tableau `jsonpaths` doit correspondre à celui des colonnes de la table cible ou de la liste de colonnes, si une liste de colonnes est utilisée.

Pour des exemples montrant comment charger des données à l'aide de l'`'auto'` argument ou d'un JSONPaths fichier, et à l'aide d'objets ou de tableaux JSON, consultezCopier à partir d'exemples JSON.

Pour plus d'informations sur la copie de plusieurs fichiers JSON, consultez Utilisation d'un manifeste pour spécifier les fichiers de données.

Caractères d'échappement dans JSON

La commande COPY charge `\n` comme un caractère de saut de ligne et charge `\t` comme un caractère de tabulation. Pour charger une barre oblique inverse, précédez-la d'une barre oblique inverse (`\\`).

Par exemple, supposons que le JSON suivant se trouve dans un fichier nommé `escape.json` dans le compartiment `s3://amzn-s3-demo-bucket/json/`.

```
{
  "backslash": "This is a backslash: \\",
  "newline": "This sentence\n is on two lines.",
  "tab": "This sentence \t contains a tab."
}
```

Exécutez les commandes suivantes pour créer la table ESCAPES et charger le fichier JSON.

```
create table escapes (backslash varchar(25), newline varchar(35), tab varchar(35));

copy escapes from 's3://amzn-s3-demo-bucket/json/escape.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as json 'auto';
```

Interrogez la table ESCAPES pour afficher les résultats.

```
select * from escapes;

       backslash        |      newline      |               tab
------------------------+-------------------+-----------------------------------
 This is a backslash: \ | This sentence     | This sentence    contains a tab.
                        :  is on two lines.
(1 row)
```

Perte de précision numérique

Vous pouvez perdre de la précision lors du chargement de nombres à partir de fichiers de données en format JSON dans une colonne qui est définie comme un type de données numérique. Certaines valeurs en virgule flottantes ne seront pas représentées de manière exacte sur les systèmes informatiques. De ce fait, des données que vous copiez depuis un fichier JSON sont susceptibles de ne pas être arrondies comme vous vous y attendez. Pour éviter une perte de précision, nous recommandons l'utilisation d'une des autres solutions suivantes :

- Représenter le nombre sous la forme d'une chaîne en plaçant la valeur entre guillemets doubles.
- Utiliser ROUNDEC pour arrondir le nombre au lieu de le tronquer.
- Plutôt que des fichiers JSON ou Avro, utiliser des fichiers CSV, délimités par un caractère, ou des fichiers texte à largeur fixe.

COPY depuis les formats de données en colonnes

COPY peut charger les données depuis Amazon S3 dans les formats en colonnes suivants :

- ORC
- Parquet

Pour des exemples d'utilisation de la commande COPY à partir de formats de données en colonnes, consultez Exemples de commandes COPY.

COPY prend en charge les données formatées en colonnes avec les considérations suivantes :

- Le compartiment Amazon S3 doit se trouver dans la même AWS région que la base de données Amazon Redshift.
- Pour accéder à vos données Amazon S3 via un point de terminaison de VPC, configurez l'accès à l'aide de politiques IAM et de rôles IAM, comme décrit dans Utilisation d'Amazon Redshift Spectrum avec le routage VPC amélioré dans le Guide de gestion Amazon Redshift.
- La commande COPY n'applique pas automatiquement l'encodage de compression.
- Seuls les paramètres COPY suivants sont pris en charge :
  - ACCEPTINVCHARS lors de la copie à partir d'un fichier ORC ou Parquet.
  - FILLRECORD
  - FROM
  - IAM_ROLE
  - CREDENTIALS
  - STATUPDATE
  - MANIFEST
  - EXPLICIT_IDS
- Si la commande COPY rencontre une erreur lors du chargement, la commande échoue. ACCEPTANYDATE et MAXERROR ne sont pas pris en charge pour les types de données en colonnes.
- Les messages d'erreur sont envoyés au client SQL. Certaines erreurs sont consignées dans STL_LOAD_ERRORS et STL_ERROR.
- La commande COPY insère les valeurs dans les colonnes de la table cible dans le même ordre que celui où les colonnes se trouvent dans les fichiers de données en colonnes. Le nombre de colonnes de la table cible et le nombre de colonnes du fichier de données doivent correspondre.

- Si le fichier que vous spécifiez pour l'opération COPY inclut l'une des extensions ci-après, nous décompressons les données sans avoir besoin d'ajouter des paramètres :
  - `.gz`
  - `.snappy`
  - `.bz2`
- La commande COPY à partir des formats de fichiers Parquet et ORC utilise Redshift Spectrum et l'accès au compartiment. Pour utiliser COPY pour ces formats, assurez-vous qu'aucune politique IAM ne bloque l'utilisation d'Amazon S3 URLs présigné. Les fichiers présignés URLs générés par Amazon Redshift sont valides pendant 1 heure afin qu'Amazon Redshift ait suffisamment de temps pour charger tous les fichiers depuis le compartiment Amazon S3. Une URL présignée unique est générée pour chaque fichier scanné par COPY à partir de formats de données en colonnes. Pour les politiques de compartiment qui incluent une `s3:signatureAge` action, veillez à définir la valeur sur au moins 3 600 000 millisecondes. Pour plus d'informations, consultez Utilisation d'Amazon Redshift Spectrum avec le routage VPC amélioré.
- Le paramètre REGION n'est pas pris en charge avec COPY à partir de formats de données colonnaires. Même si votre compartiment Amazon S3 et votre base de données se trouvent dans le même emplacement Région AWS, vous pouvez rencontrer une erreur, telle que l'argument REGION n'est pas pris en charge pour le COPY basé sur PARQUET.

Chaînes DATEFORMAT et TIMEFORMAT

La commande COPY utilise les options DATEFORMAT et TIMEFORMAT pour analyser les valeurs de date et d'heure de vos données sources. DATEFORMAT et TIMEFORMAT sont des chaînes formatées qui doivent correspondre au format des valeurs de date et d'heure de vos données source. Par exemple, une commande COPY chargeant des données source avec la valeur de date `Jan-01-1999` doit inclure la chaîne DATEFORMAT suivante :

```
COPY ...
            DATEFORMAT AS 'MON-DD-YYYY'
```

Pour plus d'informations sur la gestion des conversions de données COPY, consultez Paramètres de conversion de données.

Les chaînes DATEFORMAT et TIMEFORMAT peuvent contenir des séparateurs date/heure (tels que « `-` », « `/` » ou « `:` »), ainsi que les formats datepart et timepart présentés dans la table suivante.

> **Note**
>
> Si vous ne pouvez pas faire correspondre le format de vos valeurs de date ou d'heure avec les dateparts et timeparts suivants, ou si vos valeurs de date et d'heure utilisent des formats différents les uns des autres, utilisez l'argument `'auto'` avec le paramètre DATEFORMAT ou TIMEFORMAT. L'argument `'auto'` reconnaît plusieurs formats qui ne sont pas pris en charge lors de l'utilisation d'une chaîne DATEFORMAT ou TIMEFORMAT. Pour de plus amples informations, veuillez consulter Utilisation de la reconnaissance automatique avec DATEFORMAT et TIMEFORMAT.

Partie de date ou d'horodatage	Signification
YY	Année sans siècle
YYYY	Année avec siècle
MM	Mois en tant que nombre
MON	Mois en tant que un nom (nom abrégé ou nom complet)
DD	Jour du mois en tant que nombre
HH ou HH24	Heure (24 heures) **Note** Dans les chaînes au format DATETIME pour les fonctions SQL, HH est identique à. HH12 Toutefois, dans les chaînes DATEFORMAT et TIMEFORMAT pour COPY, HH est identique à. HH24
HH12	Heure (12 heures)
MI	Minutes

Partie de date ou d’horodatage	Signification
SS	Secondes
AM ou PM	Indicateur méridien (pour 12 heures)

Le format de date par défaut est YYYY-MM-DD. Le format d'horodatage sans fuseau horaire (TIMESTAMP) par défaut est HH:MI:SS. YYYY-MM-DD L'horodatage par défaut avec fuseau horaire (TIMESTAMPTZ) est YYYY-MM-DD HH:MI:SSOF, où OF est le décalage par rapport à l'UTC (par exemple, - 8:00. Vous ne pouvez pas inclure de spécificateur de fuseau horaire (TZ, tz ou OF) dans le timeformat_string. Le champ des secondes (SS) prend également en charge les fractions de secondes jusqu'à un niveau de détail de la microseconde. Pour charger les données TIMESTAMPTZ qui sont dans un format différent du format par défaut, spécifiez « auto ».

Vous trouverez ci-dessous des exemples de dates ou d’heures que vous pouvez trouver dans vos données sources, ainsi que les chaînes DATEFORMAT ou TIMEFORMAT correspondantes.

Exemple de date ou d’heure des données sources	Syntaxe DATEFORMAT ou TIMEFORMAT
03/31/2003	FORMAT DE DATE SOUS LA FORME « » MM/DD/YYYY
31 mars 2003	DATEFORMAT AS ’MON DD, YYYY’
03.31.2003 18:45:05 03.31.2003 18:45:05.123456	TIMEFORMAT AS ’MM.DD.YYYY HH:MI:SS’

exemple

Pour obtenir un exemple d’utilisation de TIMEFORMAT, consultez Charger un horodatage ou une datation.

Utilisation de la reconnaissance automatique avec DATEFORMAT et TIMEFORMAT

Si vous spécifiez `'auto'` comme argument pour le paramètre DATEFORMAT ou TIMEFORMAT, Amazon Redshift détecte et convertit automatiquement le format de date ou d'heure de vos données sources. Vous en trouverez un exemple ci-dessous.

```
copy favoritemovies from 'dynamodb://ProductCatalog'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
dateformat 'auto';
```

Lorsqu'elle est utilisée avec l'argument `'auto'` pour DATEFORMAT et TIMEFORMAT, la commande COPY reconnaît et convertit les formats de date et d'heure répertoriés dans la table de Chaînes DATEFORMAT et TIMEFORMAT. En outre, l'argument `'auto'` reconnaît les formats suivants qui ne sont pas pris en charge lors de l'utilisation d'une chaîne DATEFORMAT et TIMEFORMAT.

Format	Exemple de chaîne d'entrée valide
ISO 8601	2019-02-11T05:09:12.195Z
Julian	J2451187
BC	Jan-08-95 BC
YYYYMMDD HHMISS	19960108 040809
YYMMDD HHMISS	960108 040809
YYYY.DDD	1996.008
YYYY-MM-DD HH : MI : SS.SSS	1996-01-08 04:05:06.789
JJ lun HH:MI:SS AAAA TZ	17 déc 07:37:16 1997 PST
MM/DD/YYYYHH : MI : SS.SS TZ	12/17/1997 07:37:16.00 PST
YYYY-MM-DD HH : MI : SS +/-TZ	1997-12-17 07:37:16-08

Format	Exemple de chaîne d'entrée valide
JJ.MM.AAAA HH:MI:SS TZ	12.17.1997 07:37:16.00 PST

La reconnaissance automatique ne prend pas en charge epochsecs et epochmillisecs.

Pour vérifier si une valeur de date ou d'horodatage est automatiquement convertie, utilisez une fonction CAST pour tenter de convertir la chaîne en une valeur de date ou d'horodatage. Par exemple, les commandes suivantes testent la valeur d'horodatage `'J2345678 04:05:06.789'` :

```
create table formattest (test char(21));
insert into formattest values('J2345678 04:05:06.789');
select test, cast(test as timestamp) as timestamp, cast(test as date) as date from
 formattest;

        test          |      timestamp      | date
----------------------+---------------------+------------
J2345678 04:05:06.789   1710-02-23 04:05:06 1710-02-23
```

Si les données source d'une colonne DATE comprennent des informations sur l'heure, le composant est tronqué. Si les données source d'une colonne TIMESTAMP omettent des informations sur l'heure, 00:00:00 est utilisé comme composant d'heure.

Exemples de commandes COPY

> **Note**
>
> Ces exemples contiennent des sauts de ligne pour faciliter la lecture. N'incluez pas de sauts de ligne, ni d'espaces dans votre chaîne credentials-args.

Rubriques

- Charger FAVORITEMOVIES depuis une table DynamoDB
- Charger la table LISTING depuis un compartiment Amazon S3
- Charger la table LISTING depuis un cluster Amazon EMR
- Utilisation d'un manifeste pour spécifier les fichiers de données
- Charger la table LISTING à partir d'un fichier séparés par une barre verticale (délimiteur par défaut)

- Charger LISTING en utilisant des données en colonnes en format Parquet
- Chargement du LISTING à l'aide de données en colonnes au format ORC
- Charger la table EVENT avec des options
- Charger la table VENUE à partir d'un fichier de données de largeur fixe
- Charger la table CATEGORY à partir d'un fichier CSV
- Charger la table VENUE avec des valeurs EXPLICIT pour une colonne IDENTITY
- Charger la table TIME à partir d'un fichier GZIP séparé par une barre verticale
- Charger un horodatage ou une datation
- Charger les données d'un fichier avec des valeurs par défaut
- Exécuter la commande COPY des données avec l'option ESCAPE
- Copier à partir d'exemples JSON
- Copier depuis des exemples Avro
- Préparation de fichiers pour la commande COPY avec l'option ESCAPE
- Chargement d'un shapefile dans Amazon Redshift
- Commande COPY avec l'option NOLOAD
- Commande COPY avec un délimiteur multioctet et l'option ENCODING

Charger FAVORITEMOVIES depuis une table DynamoDB

Ils AWS SDKs incluent un exemple simple de création d'une table DynamoDB appelée Movies. (Pour cet exemple, consultez Mise en route avec DynamoDB.) L'exemple suivant charge la table Amazon Redshift MOVIES avec les données provenant de la table DynamoDB. La table Amazon Redshift doit déjà exister dans la base de données.

```
copy favoritemovies from 'dynamodb://Movies'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
readratio 50;
```

Charger la table LISTING depuis un compartiment Amazon S3

L'exemple suivant charge la table LISTING depuis un compartiment Amazon S3. La commande COPY charge tous les fichiers dans le dossier `/data/listing/`.

```
copy listing
from 's3://amzn-s3-demo-bucket/data/listing/'
```

```
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Charger la table LISTING depuis un cluster Amazon EMR

L'exemple suivant charge la table SALES avec les données délimités par des tabulations des fichiers compressés lzop dans un cluster Amazon EMR. La commande COPY charge tous les fichiers du dossier `myoutput/` qui commencent par `part-`.

```
copy sales
from 'emr://j-SAMPLE2B500FC/myoutput/part-*'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '\t' lzop;
```

L'exemple suivant charge la table SALES avec des données au format JSON dans un cluster Amazon EMR. La commande COPY charge tous les fichiers du dossier `myoutput/json/`.

```
copy sales
from 'emr://j-SAMPLE2B500FC/myoutput/json/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
JSON 's3://amzn-s3-demo-bucket/jsonpaths.txt';
```

Utilisation d'un manifeste pour spécifier les fichiers de données

Vous pouvez utiliser un manifeste pour vous assurer que votre commande COPY charge tous les fichiers requis et uniquement les fichiers requis, à partir d'Amazon S3. Vous pouvez également utiliser un manifeste lorsque vous avez besoin de charger plusieurs fichiers de différents compartiments ou des fichiers qui ne partagent pas le même préfixe.

Par exemple, supposons que vous devez charger les trois fichiers suivants : `custdata1.txt`, `custdata2.txt` et `custdata3.txt`. Vous pouvez utiliser la commande suivante pour charger tous les fichiers qui commencent par `amzn-s3-demo-bucket` dans `custdata` en spécifiant un préfixe :

```
copy category
from 's3://amzn-s3-demo-bucket/custdata'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

S'il n'existe que deux des fichiers en raison d'une erreur, la commande COPY charge uniquement ces deux fichiers et se termine correctement, ce qui entraîne une charge de données incomplète. Si le compartiment contient également un fichier indésirable qui utilise le même préfixe, par exemple, un

fichier nommé `custdata.backup`, la commande COPY charge ce fichier également, ce qui entraîne le chargement de données indésirables.

Pour vous assurer que tous les fichiers nécessaires sont chargés et pour empêcher que des fichiers indésirables soient chargés, vous pouvez utiliser un fichier manifeste. Le manifeste est un fichier texte au format JSON qui répertorie les fichiers à traiter par la commande COPY. Par exemple, le manifeste suivant charge les trois fichiers dans l'exemple précédent.

```
{
   "entries":[
      {
         "url":"s3://amzn-s3-demo-bucket/custdata.1",
         "mandatory":true
      },
      {
         "url":"s3://amzn-s3-demo-bucket/custdata.2",
         "mandatory":true
      },
      {
         "url":"s3://amzn-s3-demo-bucket/custdata.3",
         "mandatory":true
      }
   ]
}
```

L'indicateur `mandatory` facultatif indique si la commande COPY doit s'arrêter si le fichier n'existe pas. La valeur par défaut est `false`. Quels que soient les paramètres obligatoires, la commande COPY s'arrête si aucun fichier n'est trouvé. Dans cet exemple, la commande COPY renvoie une erreur si l'un des fichiers est introuvable. Les fichiers indésirables qui peuvent avoir été collectés si vous avez spécifié uniquement un préfixe de clé, comme `custdata.backup`, sont ignorés, car ils ne sont pas sur le manifeste.

Lors du chargement des fichiers de données au format ORC or Parquet, le champ `meta` est obligatoire, comme illustré dans l'exemple suivant.

```
{
   "entries":[
      {
         "url":"s3://amzn-s3-demo-bucket1/orc/2013-10-04-custdata",
         "mandatory":true,
         "meta":{
```

```
            "content_length":99
         }
      },
      {
         "url":"s3://amzn-s3-demo-bucket2/orc/2013-10-05-custdata",
         "mandatory":true,
         "meta":{
            "content_length":99
         }
      }
   ]
}
```

L'exemple suivant utilise un manifeste nommé `cust.manifest`.

```
copy customer
from 's3://amzn-s3-demo-bucket/cust.manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as orc
manifest;
```

Vous pouvez utiliser un manifeste pour charger les fichiers de différents compartiments ou les fichiers qui ne partagent pas le même préfixe. L'exemple suivant illustre le format JSON permettant de charger des données avec des fichiers dont les noms commencent par un horodatage.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/2013-10-04-custdata.txt","mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket/2013-10-05-custdata.txt","mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket/2013-10-06-custdata.txt","mandatory":true},
    {"url":"s3://amzn-s3-demo-bucket/2013-10-07-custdata.txt","mandatory":true}
  ]
}
```

Le manifeste peut répertorier les fichiers qui se trouvent dans différents compartiments, à condition que les compartiments se trouvent dans la même AWS région que le cluster.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket1/custdata1.txt","mandatory":false},
    {"url":"s3://amzn-s3-demo-bucket2/custdata1.txt","mandatory":false},
```

```
    {"url":"s3://amzn-s3-demo-bucket2/custdata2.txt","mandatory":false}
  ]
}
```

Charger la table LISTING à partir d'un fichier séparés par une barre verticale (délimiteur par défaut)

L'exemple suivant est un cas très simple dans lequel aucune option n'est spécifiée et le fichier d'entrée contient le délimiteur par défaut, une barre verticale ('|').

```
copy listing
from 's3://amzn-s3-demo-bucket/data/listings_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Charger LISTING en utilisant des données en colonnes en format Parquet

L'exemple suivant charge des données depuis un dossier sur Amazon S3 nommé parquet.

```
copy listing
from 's3://amzn-s3-demo-bucket/data/listings/parquet/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as parquet;
```

Chargement du LISTING à l'aide de données en colonnes au format ORC

L'exemple suivant charge des données depuis un dossier sur Amazon S3 nommé `orc`.

```
copy listing
from 's3://amzn-s3-demo-bucket/data/listings/orc/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as orc;
```

Charger la table EVENT avec des options

L'exemple suivant charge des données séparées par une barre verticale dans la table EVENT et applique les règles suivantes :

- Si des paires de guillemets anglais sont utilisées pour entourer des chaînes de caractères, elles sont supprimées.
- Les chaînes vides et les chaînes qui contiennent des espaces vides sont chargées en tant que valeurs NULL.

- La charge échoue si plus de 5 erreurs sont renvoyées.
- Les valeurs d'horodatage doivent être conformes au format spécifié ; par exemple, un horodatage valide est `2008-09-26 05:43:12`.

```
copy event
from 's3://amzn-s3-demo-bucket/data/allevents_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
removequotes
emptyasnull
blanksasnull
maxerror 5
delimiter '|'
timeformat 'YYYY-MM-DD HH:MI:SS';
```

Charger la table VENUE à partir d'un fichier de données de largeur fixe

```
copy venue
from 's3://amzn-s3-demo-bucket/data/venue_fw.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth 'venueid:3,venuename:25,venuecity:12,venuestate:2,venueseats:6';
```

L'exemple précédent suppose un fichier de données au format identique à celui des exemples de données affichés. Dans le prochain exemple, les espaces se comportent comme des espaces réservés afin que toutes les colonnes soient de la même largeur, tel qu'indiqué dans la spécification :

```
1  Toyota Park              Bridgeview  IL0
2  Columbus Crew Stadium    Columbus    OH0
3  RFK Stadium              Washington  DC0
4  CommunityAmerica BallparkKansas City KS0
5  Gillette Stadium         Foxborough  MA68756
```

Charger la table CATEGORY à partir d'un fichier CSV

Supposons que vous voulez charger la table CATEGORY avec les valeurs indiquées dans le tableau suivant.

catid	catgroup	catName	catdesc
12	Shows	Musicals	Musical theatre

catid	catgroup	catName	catdesc
13	Shows	Plays	All "non-musical" theatre
14	Shows	Opera	All opera, light, and "rock" opera
15	Concerts	Classical	All symphony, concerto, and choir concerts

L'exemple suivant montre le contenu d'un fichier texte avec les champs de valeurs séparés par des virgules.

```
12,Shows,Musicals,Musical theatre
13,Shows,Plays,All "non-musical" theatre
14,Shows,Opera,All opera, light, and "rock" opera
15,Concerts,Classical,All symphony, concerto, and choir concerts
```

Si vous chargez le fichier à l'aide du paramètre DELIMITER pour spécifier des entrées séparées par des virgules, la commande COPY échoue parce que certains champs d'entrée contiennent des virgules. Vous pouvez éviter ce problème en utilisant le paramètre CSV et en entourant les champs qui contiennent des virgules de guillemets. Si le guillemet s'affiche au sein d'une chaîne entre guillemets, vous devez doubler le guillemet. Le guillemet par défaut est double. Vous devez donc ajouter à tous les guillemets doubles des guillemets doubles supplémentaires. Votre nouveau fichier d'entrée ressemble à quelque chose comme ça.

```
12,Shows,Musicals,Musical theatre
13,Shows,Plays,"All ""non-musical"" theatre"
14,Shows,Opera,"All opera, light, and ""rock"" opera"
15,Concerts,Classical,"All symphony, concerto, and choir concerts"
```

En supposant que le nom du fichier est `category_csv.txt`, vous pouvez charger le fichier en utilisant la commande COPY suivante :

```
copy category
from 's3://amzn-s3-demo-bucket/data/category_csv.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
csv;
```

Sinon, pour éviter d'avoir à doubler les guillemets doubles de votre entrée, vous pouvez spécifier un guillemet différent à l'aide du paramètre QUOTE AS. Par exemple, la version suivante de `category_csv.txt` utilise '%' comme caractère de citation :

```
12,Shows,Musicals,Musical theatre
13,Shows,Plays,%All "non-musical" theatre%
14,Shows,Opera,%All opera, light, and "rock" opera%
15,Concerts,Classical,%All symphony, concerto, and choir concerts%
```

La commande COPY suivante utilise QUOTE AS pour charger `category_csv.txt` :

```
copy category
from 's3://amzn-s3-demo-bucket/data/category_csv.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
csv quote as '%';
```

Charger la table VENUE avec des valeurs EXPLICIT pour une colonne IDENTITY

L'exemple suivant suppose que lorsque la table VENUE a été créée, au moins une colonne (telle que la colonne `venueid`) a été désignée comme colonne IDENTITY. Cette commande se substitue au comportement IDENTITY par défaut pour la génération automatique de valeurs pour une colonne IDENTITY et charge à la place des valeurs explicites depuis le fichier venue.txt. Amazon Redshift ne vérifie pas si des valeurs IDENTITY dupliquées sont chargées dans la table lors de l'utilisation de l'option EXLICIT_IDS.

```
copy venue
from 's3://amzn-s3-demo-bucket/data/venue.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
explicit_ids;
```

Charger la table TIME à partir d'un fichier GZIP séparé par une barre verticale

L'exemple suivant charge la table TIME à partir d'un fichier GZIP séparé par une barre verticale :

```
copy time
from 's3://amzn-s3-demo-bucket/data/timerows.gz'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
gzip
delimiter '|';
```

Charger un horodatage ou une datation

L'exemple suivant charge les données avec un horodatage formaté.

> **Note**
>
> Le TIMEFORMAT de `HH:MI:SS` peut également prendre en charge des fractions de secondes au-delà de SS jusqu'aux microsecondes. Le fichier `time.txt` utilisé dans cet exemple contient une seule ligne, `2009-01-12 14:15:57.119568`.

```
copy timestamp1
from 's3://amzn-s3-demo-bucket/data/time.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
timeformat 'YYYY-MM-DD HH:MI:SS';
```

Le résultat de cette copie est le suivant :

```
select * from timestamp1;
c1
----------------------------
2009-01-12 14:15:57.119568
(1 row)
```

Charger les données d'un fichier avec des valeurs par défaut

L'exemple suivant utilise une variation de la table VENUE dans la base de données TICKIT. Prenons une table VENUE_NEW définie avec l'instruction suivante :

```
create table venue_new(
venueid smallint not null,
venuename varchar(100) not null,
venuecity varchar(30),
venuestate char(2),
venueseats integer not null default '1000');
```

Imaginons un fichier de données venue_noseats.txt qui ne contient aucune valeur pour la colonne VENUESEATS, comme illustré dans l'exemple suivant :

```
1|Toyota Park|Bridgeview|IL|
2|Columbus Crew Stadium|Columbus|OH|
```

```
3|RFK Stadium|Washington|DC|
4|CommunityAmerica Ballpark|Kansas City|KS|
5|Gillette Stadium|Foxborough|MA|
6|New York Giants Stadium|East Rutherford|NJ|
7|BMO Field|Toronto|ON|
8|The Home Depot Center|Carson|CA|
9|Dick's Sporting Goods Park|Commerce City|CO|
10|Pizza Hut Park|Frisco|TX|
```

L'instruction COPY suivante charge correctement la table depuis le fichier et applique la valeur DEFAULT ('1000') à la colonne omise :

```
copy venue_new(venueid, venuename, venuecity, venuestate)
from 's3://amzn-s3-demo-bucket/data/venue_noseats.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|';
```

Maintenant, affichez la table chargée :

```
select * from venue_new order by venueid;
venueid |         venuename          |    venuecity    | venuestate | venueseats
---------+----------------------------+-----------------+------------+------------
1 | Toyota Park                | Bridgeview      | IL         |       1000
2 | Columbus Crew Stadium      | Columbus        | OH         |       1000
3 | RFK Stadium                | Washington      | DC         |       1000
4 | CommunityAmerica Ballpark  | Kansas City     | KS         |       1000
5 | Gillette Stadium           | Foxborough      | MA         |       1000
6 | New York Giants Stadium    | East Rutherford | NJ         |       1000
7 | BMO Field                  | Toronto         | ON         |       1000
8 | The Home Depot Center      | Carson          | CA         |       1000
9 | Dick's Sporting Goods Park | Commerce City   | CO         |       1000
10 | Pizza Hut Park             | Frisco          | TX         |       1000
(10 rows)
```

Pour l'exemple suivant, en plus de supposer qu'aucune donnée VENUESEATS n'est incluse dans le fichier, supposons également qu'aucune donnée VENUENAME n'est incluse :

```
1||Bridgeview|IL|
2||Columbus|OH|
3||Washington|DC|
4||Kansas City|KS|
5||Foxborough|MA|
```

```
6||East Rutherford|NJ|
7||Toronto|ON|
8||Carson|CA|
9||Commerce City|CO|
10||Frisco|TX|
```

A l'aide de la même définition de table, l'instructions COPY suivante échoue, car aucune valeur DEFAULT n'a été spécifié pour VENUENAME et VENUENAME est une colonne NOT NULL :

```
copy venue(venueid, venuecity, venuestate)
from 's3://amzn-s3-demo-bucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|';
```

Maintenant, prenons une variation de la table VENUE qui utilise une colonne IDENTITY :

```
create table venue_identity(
venueid int identity(1,1),
venuename varchar(100) not null,
venuecity varchar(30),
venuestate char(2),
venueseats integer not null default '1000');
```

Comme dans l'exemple précédent, supposons que la colonne VENUESEATS n'a aucune valeur correspondante dans le fichier source. L'instruction COPY suivante charge la table avec succès, y compris les valeurs de données IDENTITY prédéfinies au lieu de générer ces valeurs :

```
copy venue(venueid, venuename, venuecity, venuestate)
from 's3://amzn-s3-demo-bucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' explicit_ids;
```

Cette instruction échoue, car elle n'inclut pas la colonne IDENTITY (VENUEID est manquante dans la liste de colonnes) mais inclut un paramètre EXPLICIT_IDS :

```
copy venue(venuename, venuecity, venuestate)
from 's3://amzn-s3-demo-bucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' explicit_ids;
```

Cette instruction échoue, car elle n'inclut pas un paramètre EXPLICIT_IDS :

```
copy venue(venueid, venuename, venuecity, venuestate)
from 's3://amzn-s3-demo-bucket/data/venue_pipe.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|';
```

Exécuter la commande COPY des données avec l'option ESCAPE

L'exemple suivant présente le chargement des caractères qui correspondent au délimiteur (dans ce cas, la barre verticale). Dans le fichier d'entrée, assurez-vous qu'une barre oblique inverse (\) est ajoutée à toutes les barres verticales (|) que vous voulez charger. Puis chargez le fichier avec le paramètre ESCAPE.

```
$ more redshiftinfo.txt
1|public\|event\|dwuser
2|public\|sales\|dwuser

create table redshiftinfo(infoid int,tableinfo varchar(50));

copy redshiftinfo from 's3://amzn-s3-demo-bucket/data/redshiftinfo.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter '|' escape;

select * from redshiftinfo order by 1;
infoid |       tableinfo
-------+--------------------
1      | public|event|dwuser
2      | public|sales|dwuser
(2 rows)
```

Sans le paramètre ESCAPE, cette commande COPY échoue avec une erreur `Extra column(s) found`.

> **Important**
>
> Si vous chargez vos données à l'aide d'une commande COPY avec le paramètre ESCAPE, vous devez également spécifier le paramètre ESCAPE avec votre commande UNLOAD pour générer le fichier de sortie réciproque. De même, si vous exécuter la commande UNLOAD à l'aide du paramètre ESCAPE, vous devez utiliser la commande ESCAPE lorsque vous exécutez la commande COPY sur les mêmes données.

Copier à partir d’exemples JSON

Dans les exemples suivants, vous chargez la table CATEGORY avec les données suivantes.

CATID	CATGROUP	CATNAME	CATDESC
1	Sport	MLB	Major League Baseball
2	Sport	NHL	National Hockey League
3	Sport	NFL	National Football League
4	Sport	NBA	National Basketball Association
5	Concerts	Classical	All symphony, concerto, and choir concerts

Rubriques

- Charger des données JSON à l’aide de l’option ’auto’
- Charger des données JSON à l’aide de l’option ’auto ignorecase’
- Charger à partir de données JSON à l'aide d'un JSONPaths fichier
- Chargement à partir de tableaux JSON à l'aide d'un JSONPaths fichier

Charger des données JSON à l’aide de l’option ’auto’

Pour charger des données JSON à l’aide de l’option `'auto'`, les données JSON doivent consister en un ensemble d’objets. Les noms de clés doivent correspondre aux noms de colonnes, mais dans ce cas, l’ordre n’a pas d’importance. Ce qui suit montre le contenu d’un fichier nommé `category_object_auto.json`.

```
{
    "catdesc": "Major League Baseball",
    "catid": 1,
    "catgroup": "Sports",
    "catname": "MLB"
}
{
    "catgroup": "Sports",
```

```
    "catid": 2,
    "catname": "NHL",
    "catdesc": "National Hockey League"
}
{
    "catid": 3,
    "catname": "NFL",
    "catgroup": "Sports",
    "catdesc": "National Football League"
}
{
    "bogus": "Bogus Sports LLC",
    "catid": 4,
    "catgroup": "Sports",
    "catname": "NBA",
    "catdesc": "National Basketball Association"
}
{
    "catid": 5,
    "catgroup": "Shows",
    "catname": "Musicals",
    "catdesc": "All symphony, concerto, and choir concerts"
}
```

Pour charger depuis le fichier de données JSON dans l'exemple précédent, exécutez la commande COPY suivante.

```
copy category
from 's3://amzn-s3-demo-bucket/category_object_auto.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 'auto';
```

Charger des données JSON à l'aide de l'option 'auto ignorecase'

Pour charger des données JSON à l'aide de l'option `'auto ignorecase'`, les données JSON doivent consister en un ensemble d'objets. Le cas des noms de clés n'a pas besoin de correspondre aux noms de colonnes et l'ordre n'a pas d'importance. Ce qui suit montre le contenu d'un fichier nommé `category_object_auto-ignorecase.json`.

```
{
    "CatDesc": "Major League Baseball",
    "CatID": 1,
```

```
    "CatGroup": "Sports",
    "CatName": "MLB"
}
{
    "CatGroup": "Sports",
    "CatID": 2,
    "CatName": "NHL",
    "CatDesc": "National Hockey League"
}
{
    "CatID": 3,
    "CatName": "NFL",
    "CatGroup": "Sports",
    "CatDesc": "National Football League"
}
{
    "bogus": "Bogus Sports LLC",
    "CatID": 4,
    "CatGroup": "Sports",
    "CatName": "NBA",
    "CatDesc": "National Basketball Association"
}
{
    "CatID": 5,
    "CatGroup": "Shows",
    "CatName": "Musicals",
    "CatDesc": "All symphony, concerto, and choir concerts"
}
```

Pour charger depuis le fichier de données JSON dans l'exemple précédent, exécutez la commande COPY suivante.

```
copy category
from 's3://amzn-s3-demo-bucket/category_object_auto ignorecase.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 'auto ignorecase';
```

Charger à partir de données JSON à l'aide d'un JSONPaths fichier

Si les objets de données JSON ne correspondent pas directement aux noms de colonnes, vous pouvez utiliser un JSONPaths fichier pour mapper les éléments JSON aux colonnes. L'ordre n'a pas d'importance dans les données source JSON, mais l'ordre des expressions de JSONPaths fichier

doit correspondre à l'ordre des colonnes. Supposons que vous ayez le fichier de données suivant, nommé `category_object_paths.json`.

```
{
    "one": 1,
    "two": "Sports",
    "three": "MLB",
    "four": "Major League Baseball"
}
{
    "three": "NHL",
    "four": "National Hockey League",
    "one": 2,
    "two": "Sports"
}
{
    "two": "Sports",
    "three": "NFL",
    "one": 3,
    "four": "National Football League"
}
{
    "one": 4,
    "two": "Sports",
    "three": "NBA",
    "four": "National Basketball Association"
}
{
    "one": 6,
    "two": "Shows",
    "three": "Musicals",
    "four": "All symphony, concerto, and choir concerts"
}
```

Le JSONPaths fichier suivant, nommé`category_jsonpath.json`, met en correspondance les données source avec les colonnes du tableau.

```
{
    "jsonpaths": [
        "$['one']",
        "$['two']",
        "$['three']",
        "$['four']"
```

```
    ]
}
```

Pour charger depuis le fichier de données JSON dans l'exemple précédent, exécutez la commande COPY suivante.

```
copy category
from 's3://amzn-s3-demo-bucket/category_object_paths.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 's3://amzn-s3-demo-bucket/category_jsonpath.json';
```

Chargement à partir de tableaux JSON à l'aide d'un JSONPaths fichier

Pour charger à partir de JSON des données constituées d'un ensemble de tableaux, vous devez utiliser un JSONPaths fichier pour mapper les éléments du tableau aux colonnes. Supposons que vous ayez le fichier de données suivant, nommé `category_array_data.json`.

```
[1,"Sports","MLB","Major League Baseball"]
[2,"Sports","NHL","National Hockey League"]
[3,"Sports","NFL","National Football League"]
[4,"Sports","NBA","National Basketball Association"]
[5,"Concerts","Classical","All symphony, concerto, and choir concerts"]
```

Le JSONPaths fichier suivant, nommé`category_array_jsonpath.json`, met en correspondance les données source avec les colonnes du tableau.

```
{
    "jsonpaths": [
        "$[0]",
        "$[1]",
        "$[2]",
        "$[3]"
    ]
}
```

Pour charger depuis le fichier de données JSON dans l'exemple précédent, exécutez la commande COPY suivante.

```
copy category
```

```
from 's3://amzn-s3-demo-bucket/category_array_data.json'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
json 's3://amzn-s3-demo-bucket/category_array_jsonpath.json';
```

Copier depuis des exemples Avro

Dans les exemples suivants, vous chargez la table CATEGORY avec les données suivantes.

CATID	CATGROUP	CATNAME	CATDESC
1	Sport	MLB	Major League Baseball
2	Sport	NHL	National Hockey League
3	Sport	NFL	National Football League
4	Sport	NBA	National Basketball Association
5	Concerts	Classical	All symphony, concerto, and choir concerts

Rubriques

- Charger à partir des données Avro à l’aide de l’option ’auto’
- Charger à partir des données Avro à l’aide de l’option ’auto ignorecase’
- Charger des données depuis Avro à l'aide d'un fichier JSONPaths

Charger à partir des données Avro à l’aide de l’option ’auto’

Pour charger à partir des données Avro à l’aide de l’argument `'auto'`, les noms de champs du schéma Avro doivent correspondre aux noms de colonnes. Lorsque vous utilisez l’argument `'auto'`, l’ordre n’est pas important. Ce qui suit montre le schéma d’un fichier nommé `category_auto.avro`.

```
{
    "name": "category",
    "type": "record",
    "fields": [
        {"name": "catid", "type": "int"},
```

```
        {"name": "catdesc", "type": "string"},
        {"name": "catname", "type": "string"},
        {"name": "catgroup", "type": "string"},
}
```

Les données contenues dans un fichier Avro sont au format binaire, elles ne sont donc pas explicites. L'exemple suivant illustre une représentation JSON des données dans le fichier `category_auto.avro`.

```
{
   "catid": 1,
   "catdesc": "Major League Baseball",
   "catname": "MLB",
   "catgroup": "Sports"
}
{
   "catid": 2,
   "catdesc": "National Hockey League",
   "catname": "NHL",
   "catgroup": "Sports"
}
{
   "catid": 3,
   "catdesc": "National Basketball Association",
   "catname": "NBA",
   "catgroup": "Sports"
}
{
   "catid": 4,
   "catdesc": "All symphony, concerto, and choir concerts",
   "catname": "Classical",
   "catgroup": "Concerts"
}
```

Pour charger depuis le fichier de données Avro dans l'exemple précédent, exécutez la commande COPY suivante.

```
copy category
from 's3://amzn-s3-demo-bucket/category_auto.avro'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as avro 'auto';
```

Charger à partir des données Avro à l'aide de l'option 'auto ignorecase'

Pour charger à partir des données Avro à l'aide de l'argument `'auto ignorecase'`, la casse des noms de champs du schéma Avro ne doit pas forcément correspondre à celle des noms de colonnes. Lorsque vous utilisez l'argument `'auto ignorecase'`, l'ordre n'est pas important. Ce qui suit montre le schéma d'un fichier nommé `category_auto-ignorecase.avro`.

```
{
    "name": "category",
    "type": "record",
    "fields": [
        {"name": "CatID", "type": "int"},
        {"name": "CatDesc", "type": "string"},
        {"name": "CatName", "type": "string"},
        {"name": "CatGroup", "type": "string"},
}
```

Les données contenues dans un fichier Avro sont au format binaire, elles ne sont donc pas explicites. L'exemple suivant illustre une représentation JSON des données dans le fichier `category_auto-ignorecase.avro`.

```
{
   "CatID": 1,
   "CatDesc": "Major League Baseball",
   "CatName": "MLB",
   "CatGroup": "Sports"
}
{
   "CatID": 2,
   "CatDesc": "National Hockey League",
   "CatName": "NHL",
   "CatGroup": "Sports"
}
{
   "CatID": 3,
   "CatDesc": "National Basketball Association",
   "CatName": "NBA",
   "CatGroup": "Sports"
}
{
   "CatID": 4,
   "CatDesc": "All symphony, concerto, and choir concerts",
```

```
   "CatName": "Classical",
   "CatGroup": "Concerts"
}
```

Pour charger depuis le fichier de données Avro dans l'exemple précédent, exécutez la commande COPY suivante.

```
copy category
from 's3://amzn-s3-demo-bucket/category_auto-ignorecase.avro'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format as avro 'auto ignorecase';
```

Charger des données depuis Avro à l'aide d'un fichier JSONPaths

Si les noms de champs du schéma Avro ne correspondent pas directement aux noms de colonnes, vous pouvez utiliser un JSONPaths fichier pour mapper les éléments du schéma aux colonnes. L'ordre des expressions du JSONPaths fichier doit correspondre à l'ordre des colonnes.

Supposons que vous ayez un fichier de données nommé `category_paths.avro` qui contient les mêmes données que dans l'exemple précédent, mais avec le schéma suivant.

```
{
    "name": "category",
    "type": "record",
    "fields": [
        {"name": "id", "type": "int"},
        {"name": "desc", "type": "string"},
        {"name": "name", "type": "string"},
        {"name": "group", "type": "string"},
        {"name": "region", "type": "string"}
     ]
}
```

Le JSONPaths fichier suivant, nommé`category_path.avropath`, met en correspondance les données source avec les colonnes du tableau.

```
{
    "jsonpaths": [
        "$['id']",
        "$['group']",
        "$['name']",
```

```
        "$['desc']"
    ]
}
```

Pour charger depuis le fichier de données Avro dans l'exemple précédent, exécutez la commande COPY suivante.

```
copy category
from 's3://amzn-s3-demo-bucket/category_object_paths.avro'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
format avro 's3://amzn-s3-demo-bucket/category_path.avropath ';
```

Préparation de fichiers pour la commande COPY avec l'option ESCAPE

L'exemple suivant décrit comment préparer les données pour insérer des caractères de saut de ligne avant d'importer les données dans une table Amazon Redshift à l'aide de la commande COPY avec le paramètre ESCAPE. Si vous ne préparez pas les données afin de délimiter les caractères de saut ligne, Amazon Redshift renvoie des erreurs de charge lorsque vous exécutez la commande COPY, car le caractère de saut de ligne est généralement utilisé comme séparateur d'enregistrements.

Par exemple, prenons un fichier ou une colonne dans une table externe que vous voulez copier dans une table Amazon Redshift. Si le fichier ou la colonne contient des données de contenu au format XML ou similaires, vous devez vous assurer que tous les caractères de saut de ligne (\n) qui font partie du contenu sont insérés avec la barre oblique inverse (\).

Si un fichier ou une table contient des caractères de saut de ligne imbriqués, cela offre un modèle relativement facile à mettre en correspondance. Chaque caractère de saut de ligne imbriqué suit presque toujours un caractère > avec, potentiellement, des caractères d'espace (' ' ou une tabulation) entre les deux, comme vous pouvez le voir dans l'exemple suivant d'un fichier texte intitulé `nlTest1.txt`.

```
$ cat nlTest1.txt
<xml start>
<newline characters provide>
<line breaks at the end of each>
<line in content>
</xml>|1000
<xml>
</xml>|2000
```

Avec l'exemple suivant, vous pouvez exécuter un utilitaire de traitement de texte permettant de prétraiter le fichier source et d'insérer des caractères d'échappement si nécessaire. (Le caractère | est destiné à être utilisé comme délimiteur pour séparer les données de la colonne lorsqu'elles sont copiées dans une table Amazon Redshift.)

```
$ sed -e ':a;N;$!ba;s/>[[:space:]]*\n/>\\\n/g' nlTest1.txt > nlTest2.txt
```

De même, vous pouvez utiliser Perl pour effectuer une opération similaire :

```
cat nlTest1.txt | perl -p -e 's/>\s*\n/>\\\n/g' > nlTest2.txt
```

Pour faciliter le chargement des données à partir du fichier `nlTest2.txt` dans Amazon Redshift, nous avons créé une table à deux colonnes dans Amazon Redshift. La première colonne c1, est une colonne de caractères dont le contenu est au format XML issu du fichier `nlTest2.txt`. La deuxième colonne c2 contient des valeurs de nombres entiers chargés à partir du même fichier.

Après avoir exécuté la commande `sed`, vous pouvez charger correctement des données à partir du fichier `nlTest2.txt` dans une table Amazon Redshift à l'aide du paramètre ESCAPE.

> **Note**
>
> Lorsque vous incluez le paramètre ESCAPE avec la commande COPY, il insère un certain nombre de caractères spéciaux, parmi lesquels la barre oblique (ainsi que le saut de ligne).

```
copy t2 from 's3://amzn-s3-demo-bucket/data/nlTest2.txt'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
escape
delimiter as '|';

select * from t2 order by 2;

c1           |  c2
-------------+------
<xml start>
<newline characters provide>
<line breaks at the end of each>
<line in content>
</xml>
| 1000
```

```
<xml>
</xml>       | 2000
(2 rows)
```

Vous pouvez préparer les fichiers de données exportés à partir de bases de données externes d'une manière similaire. Par exemple, avec une base de données Oracle, vous pouvez utiliser la fonction REPLACE sur chaque colonne concernée dans une table que vous voulez copier dans Amazon Redshift.

```
SELECT c1, REPLACE(c2, \n',\\n' ) as c2 from my_table_with_xml
```

En outre, de nombreux outils d'exportation et d'extraction, de transformation, de chargement (ETL) de la base de données qui traitent régulièrement de grandes quantités de données offrent des options permettant de spécifier des caractères d'échappement et délimiteurs.

Chargement d'un shapefile dans Amazon Redshift

Les exemples suivants montrent comment charger un shapefile Esri à l'aide de la commande COPY. Pour plus d'informations sur le chargement des fichiers de forme, consultez Chargement d'un shapefile dans Amazon Redshift.

Chargement d'un shapefile

Les étapes suivantes montrent comment ingérer des OpenStreetMap données depuis Amazon S3 à l'aide de la commande COPY. Cet exemple suppose que l'archive du shapefile de Norvège provenant du site de téléchargement de Geofabrik a été chargée dans un compartiment Amazon S3 privé de votre région. AWS Les fichiers `.shp`, `.shx` et `.dbf` doivent partager le même préfixe et le même nom de fichier Amazon S3.

Intégration des données sans simplification

Les commandes suivantes créent des tables et intègrent des données qui peuvent s'adapter à la taille géométrique maximale sans aucune simplification. Ouvrez `gis_osm_natural_free_1.shp` dans votre logiciel SIG préféré et inspectez les colonnes de cette couche. Par défaut, les colonnes IDENTITY ou GEOMETRY sont les premières. Lorsque la première colonne est une colonne GEOMETRY, vous pouvez créer la table comme indiqué ci-dessous.

```
CREATE TABLE norway_natural (
   wkb_geometry GEOMETRY,
   osm_id BIGINT,
```

```
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Sinon, lorsque la première colonne est une colonne IDENTITY, vous pouvez créer la table comme indiqué ci-dessous.

```
CREATE TABLE norway_natural_with_id (
   fid INT IDENTITY(1,1),
   wkb_geometry GEOMETRY,
   osm_id BIGINT,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Maintenant, vous pouvez intégrer les données en utilisant la commande COPY.

```
COPY norway_natural FROM 's3://bucket_name/shapefiles/norway/
gis_osm_natural_free_1.shp'
FORMAT SHAPEFILE
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO: Load into table 'norway_natural' completed, 83891 record(s) loaded successfully
```

Vous pouvez également intégrer les données comme indiqué ci-dessous.

```
COPY norway_natural_with_id FROM 's3://bucket_name/shapefiles/norway/
gis_osm_natural_free_1.shp'
FORMAT SHAPEFILE
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO: Load into table 'norway_natural_with_id' completed, 83891 record(s) loaded
 successfully.
```

Intégration des données avec simplification

Les commandes suivantes créent une table et tentent d'intégrer des données qui ne rentrent pas dans la taille géométrique maximale sans aucune simplification. Inspectez le shapefile `gis_osm_water_a_free_1.shp` et créez la table appropriée comme indiqué ci-dessous.

```
CREATE TABLE norway_water (
   wkb_geometry GEOMETRY,
```

```
   osm_id BIGINT,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Lorsque la commande COPY s'exécute, une erreur est renvoyée.

```
COPY norway_water FROM 's3://bucket_name/shapefiles/norway/gis_osm_water_a_free_1.shp'
FORMAT SHAPEFILE
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
ERROR:  Load into table 'norway_water' failed.  Check 'stl_load_errors' system table
 for details.
```

L'interrogation de STL_LOAD_ERRORS indique que la géométrie est trop volumineuse.

```
SELECT line_number, btrim(colname), btrim(err_reason) FROM stl_load_errors WHERE query
 = pg_last_copy_id();
 line_number |    btrim     |                                  btrim
-------------+--------------
+-------------------------------------------------------------------------
     1184705 | wkb_geometry | Geometry size: 1513736 is larger than maximum supported
 size: 1048447
```

Pour résoudre ce problème, le paramètre SIMPLIFY AUTO est ajouté à la commande COPY afin de simplifier les géométries.

```
COPY norway_water FROM 's3://bucket_name/shapefiles/norway/gis_osm_water_a_free_1.shp'
FORMAT SHAPEFILE
SIMPLIFY AUTO
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';

INFO:  Load into table 'norway_water' completed, 1989196 record(s) loaded successfully.
```

Pour afficher les lignes et les géométries simplifiées, interrogez SVL_SPATIAL_SIMPLIFY.

```
SELECT * FROM svl_spatial_simplify WHERE query = pg_last_copy_id();
 query | line_number | maximum_tolerance | initial_size | simplified | final_size |
 final_tolerance
-------+-------------+-------------------+--------------+------------+------------
+----------------------
```

```
    20 |     1184704 |               -1 |      1513736 | t          |    1008808 |
 1.276386653895e-05
    20 |     1664115 |               -1 |      1233456 | t          |    1023584 |
 6.11707814796635e-06
```

Lorsque la commande SIMPLIY AUTO max_tolerance est utilisée avec une tolérance inférieure à celle calculée automatiquement, une erreur d'ingestion est généralement renvoyée. Dans ce cas, utilisez la commande MAXERROR pour ignorer les erreurs.

```
COPY norway_water FROM 's3://bucket_name/shapefiles/norway/gis_osm_water_a_free_1.shp'
FORMAT SHAPEFILE
SIMPLIFY AUTO 1.1E-05
MAXERROR 2
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';

INFO:  Load into table 'norway_water' completed, 1989195 record(s) loaded successfully.
INFO:  Load into table 'norway_water' completed, 1 record(s) could not be loaded.
 Check 'stl_load_errors' system table for details.
```

Interrogez à nouveau SVL_SPATIAL_SIMPLIFY pour identifier l'enregistrement que la commande COPY n'a pas réussi à charger.

```
SELECT * FROM svl_spatial_simplify WHERE query = pg_last_copy_id();
 query | line_number | maximum_tolerance | initial_size | simplified | final_size |
 final_tolerance
-------+-------------+-------------------+--------------+------------+------------
+-----------------
    29 |     1184704 |           1.1e-05 |      1513736 | f          |          0 |
          0
    29 |     1664115 |           1.1e-05 |      1233456 | t          |     794432 |
     1.1e-05
```

Dans cet exemple, le premier enregistrement n'a pas réussi à s'ajuster, c'est pourquoi la colonne `simplified` affiche false. Le deuxième enregistrement a été chargé dans la tolérance donnée. Toutefois, la taille finale est supérieure à l'utilisation de la tolérance calculée automatiquement sans spécifier la tolérance maximale.

Chargement à partir d'un shapefile compressé

La commande COPY Amazon Redshift prend en charge l'ingestion de données à partir d'un shapefile compressé. Tous les composants de shapefiles doivent avoir le même préfixe Amazon

S3 et le même suffixe de compression. Par exemple, supposons que vous souhaitiez charger les données de l'exemple précédent. Dans ce cas, les fichiers `gis_osm_water_a_free_1.shp.gz`, `gis_osm_water_a_free_1.dbf.gz` et `gis_osm_water_a_free_1.shx.gz` doivent partager le même répertoire Amazon S3. La commande COPY nécessite l'option GZIP, et la clause FROM doit spécifier le fichier compressé correct, comme indiqué ci-dessous.

```
COPY norway_natural FROM 's3://bucket_name/shapefiles/norway/compressed/
gis_osm_natural_free_1.shp.gz'
FORMAT SHAPEFILE
GZIP
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO:  Load into table 'norway_natural' completed, 83891 record(s) loaded successfully.
```

Chargement des données dans une table avec un ordre de colonnes différent

Si vous avez une table qui n'a pas `GEOMETRY` comme première colonne, vous pouvez utiliser le mappage de colonnes pour mapper des colonnes à la table cible. Par exemple, créez une table en spécifiant `osm_id` comme première colonne.

```
CREATE TABLE norway_natural_order (
   osm_id BIGINT,
   wkb_geometry GEOMETRY,
   code INT,
   fclass VARCHAR,
   name VARCHAR);
```

Intégrez ensuite un shapefile à l'aide du mappage de colonnes.

```
COPY norway_natural_order(wkb_geometry, osm_id, code, fclass, name)
FROM 's3://bucket_name/shapefiles/norway/gis_osm_natural_free_1.shp'
FORMAT SHAPEFILE
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/MyRoleName';
INFO:  Load into table 'norway_natural_order' completed, 83891 record(s) loaded
 successfully.
```

Chargement de données dans une table avec une colonne de géographies

Si vous disposez d'une table avec une colonne `GEOGRAPHY`, vous devez d'abord l'intégrer à une colonne `GEOMETRY`, puis convertir les objets en objets `GEOGRAPHY`. Par exemple, après avoir copié

votre shapefile dans une colonne GEOMETRY, modifiez le tableau pour ajouter une colonne du type de données GEOGRAPHY.

```
ALTER TABLE norway_natural ADD COLUMN wkb_geography GEOGRAPHY;
```

Convertissez ensuite les géométries en géographies.

```
UPDATE norway_natural SET wkb_geography = wkb_geometry::geography;
```

Vous pouvez éventuellement supprimer la colonne GEOMETRY.

```
ALTER TABLE norway_natural DROP COLUMN wkb_geometry;
```

Commande COPY avec l'option NOLOAD

Pour valider des fichiers de données avant de charger réellement les données, utilisez l'option NOLOAD avec la commande COPY. Amazon Redshift analyse le fichier d'entrée et affiche les erreurs éventuelles. L'exemple suivant utilise l'option NOLOAD et aucune ligne n'est réellement chargée dans la table.

```
COPY public.zipcode1
FROM 's3://amzn-s3-demo-bucket/mydata/zipcode.csv'
DELIMITER ';'
IGNOREHEADER 1 REGION 'us-east-1'
NOLOAD
CREDENTIALS 'aws_iam_role=arn:aws:iam::123456789012:role/myRedshiftRole';

Warnings:
Load into table 'zipcode1' completed, 0 record(s) loaded successfully.
```

Commande COPY avec un délimiteur multioctet et l'option ENCODING

L'exemple suivant se charge LATIN1 à partir d'un fichier Amazon S3 contenant des données multioctets. La commande COPY spécifie le délimiteur sous forme octale \302\246\303\254 pour séparer les champs du fichier d'entrée codé au format ISO-8859-1. Pour spécifier le même délimiteur en UTF-8, spécifiez. DELIMITER '¦ì'

```
COPY latin1
FROM 's3://amzn-s3-demo-bucket/multibyte/myfile'
IAM_ROLE 'arn:aws:iam::123456789012:role/myRedshiftRole'
DELIMITER '\302\246\303\254'
ENCODING ISO88591
```

CREATE DATABASE

Crée une nouvelle base de données.

Pour créer une base de données, vous devez être super-utilisateur ou disposer du privilège CREATEDB. Pour créer une base de données associée à une intégration zéro ETL, vous devez être un superutilisateur ou disposer des privilèges CREATEDB et CREATEUSER.

Vous ne pouvez pas exécuter CREATE DATABASE au sein d'un bloc de transaction (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

Syntaxe

```
CREATE DATABASE database_name
[ { [
      FROM INTEGRATION '<integration_id>'[ DATABASE '<source_database>' ]
      [ SET ]
      [ ACCEPTINVCHARS [=] { TRUE | FALSE }]
      [ QUERY_ALL_STATES [=] { TRUE | FALSE }]
      [ REFRESH_INTERVAL <interval> ]
      [ TRUNCATECOLUMNS [=] { TRUE | FALSE } ]
      [ HISTORY_MODE [=] {TRUE | FALSE} ]
    ]
    [ WITH ]
    [ OWNER [=] db_owner ]
    [ CONNECTION LIMIT { limit | UNLIMITED } ]
    [ COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE } ]
    [ ISOLATION LEVEL { SERIALIZABLE | SNAPSHOT } ]
  }
  | { FROM { { ARN '<arn>' } { WITH DATA CATALOG SCHEMA '<schema>' | WITH NO DATA
 CATALOG SCHEMA } } }
  | { IAM_ROLE  {default | 'SESSION' | 'arn:aws:iam::<account-id>:role/<role-name>' } }
  | { [ WITH PERMISSIONS ] FROM DATASHARE datashare_name OF [ ACCOUNT account_id ]
 NAMESPACE namespace_guid }
]
```

Paramètres

database_name

Nom de la nouvelle base de données. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

DEPUIS L'INTÉGRATION '<integration_id>' [BASE DE DONNÉES '<source_database>']

Spécifie s'il faut créer la base de données à l'aide d'un identifiant d'intégration zéro ETL. Vous pouvez le récupérer `integration_id` depuis la vue système SVV_INTEGRATION. Pour les intégrations Aurora PostgreSQL Zero-ETL, vous devez également `source_database` spécifier le nom, qui peut également être récupéré à partir de SVV_INTEGRATION.

Pour obtenir un exemple, consultez Créez des bases de données pour recevoir les résultats des intégrations sans ETL. Pour plus d'informations sur la création de bases de données avec des intégrations sans ETL, consultez la section Création de bases de données de destination dans Amazon Redshift dans le guide de gestion Amazon Redshift.

SET

Mot-clé facultatif.

ACCEPTINVCHARS [=] {VRAI | FAUX}

La clause ACCEPTINVCHARS indique si les tables d'intégration zéro ETL continuent à être ingérées lorsque des caractères non valides sont détectés pour le type de données VARCHAR. Lorsque des caractères non valides sont détectés, ils sont remplacés par un ? caractère par défaut.

QUERY_ALL_STATES [=] {VRAI | FAUX}

La clause QUERY_ALL_STATES indique si les tables d'intégration zéro ETL peuvent être interrogées dans tous les états (,, et). `Synced` `Failed` `ResyncRequired` `ResyncInitiated` Par défaut, une table d'intégration zéro ETL ne peut être interrogée que dans son état. `Synced`

INTERVALLE DE RAFRAÎCHISSEMENT <interval>

La clause REFRESH_INTERVAL définit l'intervalle de temps approximatif, en secondes, pour actualiser les données de la source zéro ETL vers la base de données cible. La valeur peut être définie entre 0 et 432 000 secondes (5 jours) pour les intégrations sans ETL dont le type de source est Aurora MySQL, Aurora PostgreSQL ou RDS for MySQL. Pour les intégrations Amazon DynamoDB Zero-ETL, la valeur peut être définie entre 900 et 432 000 secondes (15 minutes et 5 jours). La valeur par défaut `interval` est de zéro (0) seconde pour les intégrations sans

ETL dont le type de source est Aurora MySQL, Aurora PostgreSQL ou RDS for MySQL. Pour les intégrations Amazon DynamoDB Zero-ETL, la `interval` valeur par défaut est de 900 secondes (15 minutes).

TRONCATCOLUMNS [=] {VRAI | FAUX}

La clause TRUNCATECOLUMNS indique si les tables d'intégration Zero-ETL continuent à être ingérées lorsque les valeurs des attributs de colonne VARCHAR ou SUPER dépassent les limites. Lorsque `TRUE` les valeurs sont tronquées pour tenir dans la colonne et que les valeurs des attributs JSON débordants sont tronquées pour tenir dans la colonne SUPER.

HISTORY_MODE [=] {VRAI | FAUX}

Clause qui indique si Amazon Redshift définira le mode historique pour toutes les nouvelles tables de la base de données spécifiée. Cette option s'applique uniquement aux bases de données créées pour une intégration zéro ETL.

La clause HISTORY_MODE peut être définie sur ou. `TRUE` `FALSE` L'argument par défaut est `FALSE`. Pour plus d'informations sur HISTORY_MODE, consultez la section Mode historique dans le guide de gestion Amazon Redshift.

WITH

Mot-clé facultatif.

PROPRIÉTAIRE [=] db_owner

Spécifie le nom utilisateur du propriétaire de la base de données.

CONNECTION LIMIT { limite | UNLIMITED }

Le nombre maximum de connexions à la base de données que les utilisateurs sont autorisés à ouvrir simultanément. La limite se s'applique pas aux super-utilisateurs. Utilisez le mot-clé UNLIMITED pour autoriser le nombre maximum de connexions simultanées. Une limite sur le nombre de connexions pour chaque utilisateur peut également s'appliquer. Pour plus d'informations, consultez CREATE USER. La valeur par défaut est UNLIMITED. Pour afficher les connexions en cours, interrogez la vue système STV_SESSIONS.

> **Note**
>
> Si les deux limites de connexion (utilisateurs et base de données) s'appliquent, un emplacement de connexion inutilisé situé entre les deux limites doit également être disponible lorsqu'un utilisateur tente de se connecter.

COLLATE { CASE_SENSITIVE | CASE_INSENSITIVE }

Clause qui spécifie si la recherche ou la comparaison de chaînes est CASE_SENSITIVE (sensible à la casse) ou CASE_INSENSITIVE (insensible à la casse). La valeur par défaut est CASE_SENSITIVE.

COLLATE n'est pas pris en charge lorsque vous créez une base de données à partir d'un partage de données.

ISOLATION LEVEL { SERIALIZABLE | SNAPSHOT }

Clause qui spécifie le niveau d'isolation utilisé lorsque les requêtes sont exécutées sur une base de données.

- Isolation SÉRIALISABLE — Fournit une sérialisabilité complète pour les transactions simultanées. Pour de plus amples informations, veuillez consulter Isolement sérialisable.
- Isolation SNAPSHOT : fournit un niveau d'isolation avec protection contre les conflits de mise à jour et de suppression. Il s'agit de la valeur par défaut pour une base de données créée dans un cluster provisionné ou un espace de noms sans serveur.

Vous pouvez afficher le modèle de simultanéité exécuté par votre base de données comme suit :

- Interrogez la vue du catalogue STV_DB_ISOLATION_LEVEL. Pour de plus amples informations, veuillez consulter STV_DB_ISOLATION_LEVEL.

```
SELECT * FROM stv_db_isolation_level;
```

- Interrogez la vue PG_DATABASE_INFO.

```
SELECT datname, datconfig FROM pg_database_info;
```

Le niveau d'isolation par base de données apparaît à côté de la clé `concurrency_model`. Une valeur 1 indique SNAPSHOT. Une valeur 2 indique SERIALIZABLE.

Dans les bases de données Amazon Redshift, l'isolation SERIALIZABLE et SNAPSHOT sont des types de niveau d'isolation sérialisable. Autrement dit, les lectures corrompues, les lectures non reproductibles (non-repeatable read) et les lectures fantôme sont empêchées conformément à la norme SQL. Ces niveaux d'isolation garantissent qu'une transaction s'exécute sur un instantané de données tel qu'il existe lorsque la transaction commence, et qu'aucune autre transaction ne peut modifier cet instantané. Toutefois, l'isolation SNAPSHOT n'offre pas une sérialisation complète, car elle n'empêche pas les insertions et mises à jour asymétriques en écriture sur différentes lignes de tableau.

Le scénario suivant illustre les mises à jour asymétriques en écriture utilisant le niveau d'isolation SNAPSHOT. Une table nommée `Numbers` contient une colonne nommée `digits` qui contient les valeurs `0` et `1`. L'instruction UPDATE de chaque utilisateur ne chevauche pas l'autre utilisateur. Cependant, les valeurs `0` et `1` sont échangées. Le code SQL qu'elles exécutent suit cette chronologie avec les résultats suivants :

Heure	Action utilisateur 1	Action utilisateur 2
1	BEGIN;	
2		BEGIN;

3	SELECT * FROM Numbers;
	`digits`
	`-`
	`------`
	`0`
	`1`	

4		SELECT * FROM Numbers;
		`digits`
		`------`
		`0`
		`1`

5	UPDATE Numbers SET digits=0 WHERE	

Heure	Action utilisateur 1	Action utilisateur 2
	digits=1;	

6	SELECT * FROM Numbers
	`digits`
	`-`
	`-----`
	`0`
	`0`	

7	COMMIT	
8		Update Numbers SET digits=1 WHERE digits=0;

9		SELECT * FROM Numbers;
		`digits`
		`------`
		`1`
		`1`

10		COMMIT;

Heure	Action utilisateur 1	Action utilisateur 2

11	SELECT * FROM Numbers
	`digits`
	`-`
	`------`
	`1`
	`0`	

12		SELECT * FROM Numbers;
		`digits`
		`------`
		`1`
		`0`

Si le même scénario est exécuté à l'aide d'une isolation sérialisable, Amazon Redshift met fin à l'utilisateur 2 en raison d'une violation sérialisable et renvoie une erreur `1023`. Pour de plus amples informations, veuillez consulter Comment corriger les erreurs d'isolement sérialisable. Dans ce cas, seul l'utilisateur 1 peut s'engager avec succès. Toutes les charges de travail ne nécessitent pas une isolation sérialisable, auquel cas l'isolation des instantanés suffit comme niveau d'isolation cible pour votre base de données.

DE L'ARN « <ARN>»

L'ARN AWS Glue de base de données à utiliser pour créer la base de données.

{AVEC LE SCHÉMA DE CATALOGUE DE DONNÉES <schema>« » | SANS SCHÉMA DE CATALOGUE DE DONNÉES}

> **Note**
>
> Ce paramètre ne s'applique que si votre commande CREATE DATABASE utilise également le paramètre FROM ARN.

Indique si la base de données doit être créée à l'aide d'un schéma pour pouvoir accéder à des objets dans le AWS Glue Data Catalog.

IAM_ROLE {par défaut | 'SESSION' | 'arn:aws:iam : :role/ '} *<Compte AWS-id> <role-name>*

> **Note**
>
> Ce paramètre ne s'applique que si votre commande CREATE DATABASE utilise également le paramètre FROM ARN.

Si vous spécifiez un rôle IAM qui est associé au cluster pendant l'exécution de la commande CREATE DATABASE, Amazon Redshift utilise les informations d'identification du rôle lorsque vous exécutez des requêtes sur la base de données.

Si le mot clé `default` est spécifié, le rôle IAM défini par défaut et qui est associé au cluster est alors utilisé.

Utilisez `'SESSION'` si vous vous connectez à votre cluster Amazon Redshift à l'aide d'une identité fédérée et que vous accédez aux tables à partir du schéma externe créé à l'aide de cette commande. Pour voir un exemple d'utilisation d'une identité fédéré, consultez Utilisation d'une identité fédérée pour gérer l'accès d'Amazon Redshift aux ressources locales et aux tables externes Amazon Redshift Spectrum, qui explique comment configurer l'identité fédérée.

Utilisez l'Amazon Resource Name (ARN) d'un rôle IAM que votre cluster utilise pour l'authentification et l'autorisation. Au minimum, le rôle IAM doit être autorisé à exécuter une opération LIST sur le compartiment Amazon S3 devant être accessible et une opération GET sur les objets Amazon S3 contenus dans le compartiment. Pour en savoir plus sur l'utilisation de IAM_ROLE lors de la création d'une base de données à des AWS Glue Data Catalog fins de partage de données, consultez la section Utilisation de partages de données gérés par Lake Formation en tant que consommateur.

Le code suivant montre la syntaxe de la chaîne de paramètre IAM_ROLE pour un seul ARN.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Vous pouvez créer des chaînes de rôles pour permettre à votre cluster d'endosser un autre rôle IAM, y compris un rôle appartenant à un autre compte. Les chaînes ainsi créées peuvent inclure jusqu'à 10 rôles. Pour plus d'informations, consultez Créer des rôles IAM dans Amazon Redshift Spectrum.

Attachez à ce rôle IAM une politique d'autorisations IAM similaire à la suivante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-west-2:123456789012:secret:my-
rds-secret-VNenFy"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour connaître les étapes à suivre afin de créer un rôle IAM à utiliser avec une requête fédérée, consultez Création d'un secret et d'un rôle IAM pour utiliser des requêtes fédérées.

> **Note**
>
> N'incluez pas d'espaces dans la liste des rôles chaînés.

L'exemple suivant montre la syntaxe d'une chaîne de trois rôles.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-1-name>,arn:aws:iam::<aws-
account-id>:role/<role-2-name>,arn:aws:iam::<aws-account-id>:role/<role-3-name>'
```

Syntaxe pour l'utilisation de CREATE DATABASE avec une unité de partage des données

La syntaxe suivante décrit la commande CREATE DATABASE utilisée pour créer des bases de données à partir d'un partage de données afin de partager des données au sein d'un même AWS compte.

```
CREATE DATABASE database_name
[ [ WITH PERMISSIONS ] FROM DATASHARE datashare_name OF [ ACCOUNT account_id ]
 NAMESPACE namespace_guid
```

La syntaxe suivante décrit la commande CREATE DATABASE utilisée pour créer des bases de données à partir d'un partage de données afin de partager des données entre comptes. AWS

```
CREATE DATABASE database_name
[ [ WITH PERMISSIONS ] FROM DATASHARE datashare_name OF ACCOUNT account_id
 NAMESPACE namespace_guid
```

Paramètres pour l'utilisation de CREATE DATABASE avec une unité de partage des données

FROM DATASHARE

Mot-clé indiquant où se situe l'unité de partage des données externe.

datashare_name

Nom de la base de données sur laquelle la base de données consommateur est créée.

WITH PERMISSIONS

Spécifie que la base de données créée à partir de l'unité de partage des données nécessite des autorisations de niveau objet pour accéder à des objets de base de données individuels. Sans cette clause, les utilisateurs ou les rôles disposant de l'autorisation USAGE sur la base de données auront automatiquement accès à tous les objets de celle-ci.

NAMESPACE namespace_guid

Valeur spécifiant l'espace de noms du producteur compte auquel l'unité de partage des données appartient.

ACCOUNT account_id

Valeur spécifiant le compte du producteur auquel l'unité de partage des données appartient.

Notes d'utilisation de la commande CREATE DATABASE pour le partage de données

En tant que superutilisateur de base de données, lorsque vous utilisez CREATE DATABASE pour créer des bases de données à partir de partages de données au sein du AWS compte, spécifiez l'option NAMESPACE. L'option ACCOUNT est facultative. Lorsque vous utilisez CREATE DATABASE pour créer des bases de données provenant d'unités de partage des données sur des comptes AWS , spécifiez les options ACCOUNT et NAMESPACE du producteur.

Vous ne pouvez créer qu'une seule base de données grand public pour une unité de partage des données sur un cluster consommateur. Vous ne pouvez pas créer plusieurs bases de données grand public faisant référence à la même unité de partage des données.

CRÉER UNE BASE DE DONNÉES depuis AWS Glue Data Catalog

Pour créer une base de données à l'aide d'un ARN AWS Glue de base de données, spécifiez cet ARN dans votre commande CREATE DATABASE.

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH NO DATA CATALOG SCHEMA;
```

Vous pouvez aussi éventuellement fournir une valeur dans le paramètre IAM_ROLE. Pour en savoir plus sur le paramètre et les valeurs acceptées, consultez Paramètres.

Les exemples suivants montrent comment créer une base de données à partir d'un ARN en utilisant un rôle IAM.

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH NO DATA CATALOG SCHEMA
 IAM_ROLE <iam-role-arn>
```

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH NO DATA CATALOG SCHEMA
 IAM_ROLE default;
```

Vous pouvez également créer une base de données en utilisant un DATA CATALOG SCHEMA.

```
CREATE DATABASE sampledb FROM ARN <glue-database-arn> WITH DATA CATALOG SCHEMA
 <sample_schema> IAM_ROLE default;
```

Créez des bases de données pour recevoir les résultats des intégrations sans ETL

Pour créer une base de données à l'aide d'une identité d'intégration zéro ETL, spécifiez-la `integration_id` dans votre commande CREATE DATABASE.

```
CREATE DATABASE destination_db_name FROM INTEGRATION 'integration_id';
```

Par exemple, récupérez d'abord les identifiants d'intégration dans SVV_INTEGRATION ;

```
SELECT integration_id FROM SVV_INTEGRATION;
```

Utilisez ensuite l'un des identifiants d'intégration récupérés pour créer la base de données qui reçoit les intégrations zéro ETL.

```
CREATE DATABASE sampledb FROM INTEGRATION 'a1b2c3d4-5678-90ab-cdef-EXAMPLE11111';
```

Lorsque la base de données source des intégrations Zero-ETL est requise, spécifiez, par exemple.

```
CREATE DATABASE sampledb FROM INTEGRATION 'a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'
 DATABASE sourcedb;
```

Vous pouvez également définir un intervalle d'actualisation pour la base de données. Par exemple, pour définir l'intervalle d'actualisation à 7 200 secondes pour les données provenant d'une source d'intégration zéro ETL :

```
CREATE DATABASE myacct_mysql FROM INTEGRATION 'a1b2c3d4-5678-90ab-cdef-EXAMPLE11111'
 SET REFRESH_INTERVAL 7200;
```

Interrogez la vue du catalogue SVV_INTEGRATION pour obtenir des informations sur une intégration zéro ETL, telles que integration_id, target_database, source, refresh_interval, etc.

```
SELECT * FROM svv_integration;
```

L'exemple suivant crée une base de données à partir d'une intégration avec le mode historique activé.

```
CREATE DATABASE sample_integration_db FROM INTEGRATION 'a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111' SET HISTORY_MODE = true;
```

Limites de CREATE DATABASE

Amazon Redshift applique ces limites pour les bases de données :

- Maximum de 60 bases de données définies par l'utilisateur par cluster.
- Maximum de 127 octets pour un nom de base de données.
- Un nom de base de données ne peut pas être un mot réservé.

Classement de base de données

Le classement est un ensemble de règles qui définit la façon dont le moteur de base de données compare et trie les données de type caractère dans SQL. Le classement insensible à la casse est le classement le plus utilisé. Amazon Redshift utilise un classement insensible à la casse pour faciliter la migration à partir d'autres systèmes d'entrepôts des données. Grâce à la prise en charge native du classement insensible à la casse, Amazon Redshift continue d'utiliser des méthodes de réglage ou d'optimisation importantes, telles que les clés de distribution, les clés de tri ou l'analyse à plage restreinte.

La clause COLLATE spécifie le classement par défaut de toutes les colonnes CHAR et VARCHAR de la base de données. Si CASE_INSENSITIVE est spécifié, toutes les colonnes CHAR ou VARCHAR utilisent un classement insensible à la casse. Pour obtenir des informations sur le classement, consultez Séquences de classement.

Les données insérées ou intégrées dans des colonnes insensibles à la casse conserveront leur casse d'origine. Cependant, toutes les opérations de chaîne basées sur la comparaison, y compris le tri et le regroupement, sont insensibles à la casse. Les opérations de correspondance de modèles

telles que les prédicats LIKE, similaire à et les fonctions d'expression régulière sont également insensibles à la casse.

Les opérations SQL suivantes prennent en charge la sémantique de classement applicable :

- Opérateurs de comparaison : **=**, **<>**, **<**, **<=**, **>**, **>=**.
- Opérateur LIKE
- Clauses ORDER BY
- Clauses GROUP BY
- Fonctions d'agrégation qui utilisent la comparaison de chaînes, telles que MIN, MAX et LISTAGG
- Fonctions de fenêtrage, telles que les clauses PARTITION BY et ORDER BY
- Fonctions scalaires greatest() et least (), STRPOS(), REGEXP_COUNT (), REGEXP_REPLACE(), REGEXP_INSTR(), REGEXP_SUBSTR()
- Clause Distinct
- UNION, INTERSECT et EXCEPT
- IN LIST

Pour les requêtes externes, y compris les requêtes fédérées Amazon Redshift Spectrum et Aurora PostgreSQL, le classement de la colonne VARCHAR ou CHAR est le même que le classement actuel au niveau de la base de données.

L'exemple suivant interroge une table Amazon Redshift Spectrum :

```
SELECT ci_varchar FROM spectrum.test_collation
WHERE ci_varchar = 'AMAZON';

ci_varchar
----------
amazon
Amazon
AMAZON
AmaZon
(4 rows)
```

Pour obtenir des informations sur la création de tables à l'aide du classement de bases de données, consultez CREATE TABLE.

Pour obtenir des informations sur la fonction COLLATE, consultez Fonction COLLATE.

Limites de classement de bases de données

Les limitations suivantes concernent l'utilisation du classement de base de données dans Amazon Redshift :

- Toutes les tables ou vues système, y compris les tables catalogue PG et les tables système Amazon Redshift, sont sensibles à la casse.
- Lorsque la base de données consommateur et la base de données producteur ont des classements différents au niveau de la base de données, Amazon Redshift ne prend pas en charge les requêtes inter-bases de données et inter-clusters.
- Amazon Redshift ne prend pas en charge le classement insensible à la casse dans les requêtes au nœud principal uniquement.

  L'exemple suivant montre une requête insensible à la casse non prise en charge et l'erreur envoyée par qu'Amazon Redshift :

```
SELECT collate(usename, 'case_insensitive') FROM pg_user;
ERROR:  Case insensitive collation is not supported in leader node only query.
```

- Amazon Redshift ne prend pas en charge l'interaction entre les colonnes sensibles à la casse et non sensibles à la casse, telles que les opérations de comparaison, de fonction, de jointure ou d'ensembles.

  Les exemples suivants montrent des erreurs lorsque des colonnes sensibles à la casse et insensibles à la casse interagissent :

```
CREATE TABLE test
  (ci_col varchar(10) COLLATE case_insensitive,
   cs_col varchar(10) COLLATE case_sensitive,
   cint int,
   cbigint bigint);
```

```
SELECT ci_col = cs_col FROM test;
ERROR:  Query with different collations is not supported yet.
```

```
SELECT concat(ci_col, cs_col) FROM test;
ERROR:  Query with different collations is not supported yet.
```

```
SELECT ci_col FROM test UNION SELECT cs_col FROM test;
ERROR:  Query with different collations is not supported yet.
```

```
SELECT * FROM test a, test b WHERE a.ci_col = b.cs_col;
ERROR:  Query with different collations is not supported yet.
```

```
Select Coalesce(ci_col, cs_col) from test;
ERROR:  Query with different collations is not supported yet.
```

```
Select case when cint > 0 then ci_col else cs_col end from test;
ERROR:  Query with different collations is not supported yet.
```

- Amazon Redshift ne prend pas en charge le classement pour le type de données SUPER. La création de colonnes SUPER dans des bases de données insensibles à la casse et les interactions entre les colonnes SUPER et insensibles à la casse ne sont pas prises en charge.

  L'exemple suivant crée une table avec le type de données SUPER dans la base de données insensible à la casse :

  ```
  CREATE TABLE super_table (a super);
  ERROR: SUPER column is not supported in case insensitive database.
  ```

  L'exemple suivant interroge les données avec une chaîne insensible à la casse en les comparant avec les données SUPER :

  ```
  CREATE TABLE test_super_collation
  (s super, c varchar(10) COLLATE case_insensitive, i int);
  ```

  ```
  SELECT s = c FROM test_super_collation;
  ERROR:  Coercing from case insensitive string to SUPER is not supported.
  ```

Pour que ces requêtes fonctionnent, utilisez la fonction COLLATE afin de convertir le classement d'une colonne et la faire correspondre à l'autre. Pour plus d'informations, consultez Fonction COLLATE.

Exemples

Création d'une base de données

L'exemple suivant crée une base de données nommée TICKIT et attribue la propriété à l'utilisateur DWUSER.

```
create database tickit
with owner dwuser;
```

Interrogez la table de catalogue PG_DATABASE_INFO afin d'afficher les détails relatifs aux bases de données.

```
select datname, datdba, datconnlimit
from pg_database_info
where datdba > 1;

 datname     | datdba | datconnlimit
-------------+--------+-------------
 admin       |    100 | UNLIMITED
 reports     |    100 | 100
 tickit      |    100 | 100
```

L'exemple suivant crée une base de données nommée **`sampledb`** avec niveau d'isolation SNAPSHOT.

```
CREATE DATABASE sampledb ISOLATION LEVEL SNAPSHOT;
```

L'exemple suivant crée la base de données sales_db à partir de l'unité de partage des données salesshare.

```
CREATE DATABASE sales_db FROM DATASHARE salesshare OF NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Exemple de classement de bases de données

Création d'une base de données sensible à la casse

L'exemple suivant crée la base de données `sampledb`, crée la table T1 et insère des données dans la table T1.

```
create database sampledb collate case_insensitive;
```

Connectez-vous à la nouvelle base de données que vous venez de créer à l'aide de votre client SQL. Lorsque vous utilisez l'éditeur de requêtes Amazon Redshift, choisissez `sampledb` dans Éditeur. Lorsque vous utilisez RSQL, utilisez une commande similaire à celle-ci.

```
\connect sampledb;
```

```
CREATE TABLE T1 (
  col1 Varchar(20) distkey sortkey
);
```

```
INSERT INTO T1 VALUES ('bob'), ('john'), ('Mary'), ('JOHN'), ('Bob');
```

Ensuite, la requête trouve des résultats avec `John`.

```
SELECT * FROM T1 WHERE col1 = 'John';

 col1
 ------
 john
 JOHN
(2 row)
```

Classement par sensibilité à la casse

L'exemple suivant montre le classement par sensibilité à la casse avec la table T1. Le classement de Bob et bob ou John et john est non déterministe, car ils sont égaux dans la colonne insensible à la casse.

```
SELECT * FROM T1 ORDER BY 1;

 col1
 ------
 bob
 Bob
 JOHN
 john
```

```
 Mary
(5 rows)
```

De même, l'exemple suivant montre un classement par sensibilité à la casse avec la clause GROUP BY. Bob et bob sont égaux et appartiennent au même groupe. Lequel des deux apparaît dans le résultat est non déterministe.

```
SELECT col1, count(*) FROM T1 GROUP BY 1;

 col1 | count
 -----+------
 Mary |  1
 bob  |  2
 JOHN |  2
(3 rows)
```

Interrogation avec une fonction de fenêtrage sur des colonnes insensibles à la casse

L'exemple suivant interroge une fonction de fenêtrage sur une colonne insensible à la casse.

```
SELECT col1, rank() over (ORDER BY col1) FROM T1;

 col1 | rank
 -----+------
 bob  |   1
 Bob  |   1
 john |   3
 JOHN |   3
 Mary |   5
(5 rows)
```

Interrogation avec le mot-clé DISTINCT

L'exemple suivant interroge la table T1 avec le mot-clé DISTINCT.

```
SELECT DISTINCT col1 FROM T1;

 col1
 ------
 bob
 Mary
```

```
 john
(3 rows)
```

Interrogation avec la clause UNION

L'exemple suivant illustre les résultats de la clause UNION des tables T1 et T2.

```
CREATE TABLE T2 AS SELECT * FROM T1;
```

```
SELECT col1 FROM T1 UNION SELECT col1 FROM T2;

 col1
 ------
 john
 bob
 Mary
(3 rows)
```

CREATE DATASHARE

Crée une nouvelle unité de partage des données dans la base de données actuelle. Le propriétaire de cette unité de partage des données est l'auteur de la commande CREATE TABLE.

Amazon Redshift associe chaque unité de partage des données à une seule base de données Amazon Redshift. Vous pouvez uniquement ajouter des objets de la base de données associée à une unité de partage des données. Vous pouvez créer plusieurs unités de partage des données sur la même base de données Amazon Redshift.

Pour obtenir des informations sur les unités de partage des données, consultez Partage de données dans Amazon Redshift.

Pour afficher des informations sur les unités de partage des données, utilisez SHOW DATASHARES.

Privilèges requis

Les privilèges suivants sont requis pour CREATE DATASHARE :

- Superuser
- Utilisateurs disposant du privilège CREATE DATASHARE

- Propriétaire de la base de données

Syntaxe

```
CREATE DATASHARE datashare_name
[[SET] PUBLICACCESSIBLE [=] TRUE | FALSE ];
```

Paramètres

datashare_name

Nom du datashare. Le nom de l'unité de partage des données doit être unique dans l'espace de noms du cluster.

[[SET] PUBLICACCESSIBLE]

Clause spécifiant si l'unité de partage des données peut être partagée avec des clusters qui sont accessibles publiquement.

La valeur par défaut de `SET PUBLICACCESSIBLE` est `FALSE`.

Notes d'utilisation

Par défaut, le propriétaire de l'unité de partage des données possède le partage, mais pas les objets dans le partage.

Seuls les super-utilisateurs et le propriétaire de la base de données peuvent utiliser CREATE DATASHARE et déléguer des privilèges ALTER à d'autres utilisateurs ou groupes.

Exemples

L'exemple suivant crée l'unité de partage des données `salesshare`.

```
CREATE DATASHARE salesshare;
```

L'exemple suivant crée l'unité de partage des données `demoshare` que gère AWS Data Exchange .

```
CREATE DATASHARE demoshare SET PUBLICACCESSIBLE TRUE, MANAGEDBY ADX;
```

CREATE EXTERNAL FUNCTION

Crée une fonction scalaire définie par l'utilisateur (UDF) basée sur Amazon AWS Lambda Redshift. Pour plus d'informations sur les fonctions Lambda définies par l'utilisateur, consultez Lambda scalaire UDFs.

Privilèges requis

Les privilèges suivants sont requis pour CREATE EXTERNAL FUNCTION :

- Superuser
- Utilisateurs possédant le privilège CREATE [OR REPLACE] EXTERNAL FUNCTION

Syntaxe

```
CREATE [ OR REPLACE ] EXTERNAL FUNCTION external_fn_name ( [data_type] [, ...] )
RETURNS data_type
{ VOLATILE | STABLE }
LAMBDA 'lambda_fn_name'
IAM_ROLE { default | ‘arn:aws:iam::<Compte AWS-id>:role/<role-name>’
RETRY_TIMEOUT milliseconds
MAX_BATCH_ROWS count
MAX_BATCH_SIZE size [ KB | MB ];
```

Paramètres

OR REPLACE

Clause qui spécifie que si une fonction ayant le même nom et les mêmes types de données pour les arguments en entrée, ou signature, existe déjà, la fonction existante est remplacée. Vous pouvez uniquement remplacer une fonction par une nouvelle fonction qui définit un ensemble identique de types de données. Vous devez être un super-utilisateur pour remplacer une fonction.

Si vous définissez une fonction avec le même nom qu'une fonction existante, mais avec une signature différente, vous créez une nouvelle fonction. En d'autres termes, le nom de la fonction est surchargé. Pour plus d'informations, consultez Surcharge des noms de fonctions.

external_fn_name

Nom de la fonction externe. Si vous spécifiez un nom de schéma (tel que myschema.myfunction), la fonction est créée à l'aide du schéma spécifié. Sinon, la fonction est créée dans le schéma en cours. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

Nous vous recommandons de nommer toutes les fonctions UDF en utilisant le préfixe `f_`. Amazon Redshift réserve le préfixe `f_` pour les noms de fonctions UDF. En utilisant le préfixe `f_`, vous vous assurez que le nom de votre fonction UDF n'entrera pas en conflit avec le nom d'une fonction SQL intégrée pour Amazon Redshift, que ce soit maintenant ou à l'avenir. Pour plus d'informations, consultez Prévention des conflits de dénomination des fonctions UDF.

data_type

Type de données des arguments en entrée. Pour plus d'informations, consultez Python scalaire UDFs et Lambda scalaire UDFs.

RETURNS type_données

Type de données de la valeur renvoyée par la fonction. Le type de données RETURNS peut être n'importe quel type de données Amazon Redshift standard. Pour plus d'informations, consultez Python scalaire UDFs et Lambda scalaire UDFs.

VOLATILE | STABLE

Informe l'optimiseur de requête à propos de l'instabilité de la fonction.

Pour obtenir la meilleure optimisation possible, qualifiez votre fonction avec la catégorie d'instabilité la plus stricte qui s'y applique. En matière de rigueur, en commençant par la moins stricte, les catégories d'instabilité sont les suivantes :

- VOLATILE
- STABLE

VOLATILE

Soit les mêmes arguments, la fonction peut renvoyer des résultats différents sur des appels successifs, y compris pour les lignes d'une même instruction. L'optimiseur de requête ne peut pas émettre d'hypothèses concernant le comportement d'une fonction volatile. Une requête qui utilise une fonction volatile doit réévaluer la fonction pour chaque entrée.

STABLE

À partir des mêmes arguments, la fonction renvoie invariablement les mêmes résultats lors des appels successifs traités au sein d'une même instruction. La fonction peut renvoyer des résultats différents lorsqu'elle est appelée dans différents instructions. Cette catégorie vise ici à permettre à l'optimiseur de réduire le nombre de fois que la fonction est appelée au sein d'une même instruction.

Notez que si la rigueur choisie n'est pas valide pour la fonction, l'optimiseur risque d'ignorer certains appels basés sur cette rigueur. Cela peut produire un jeu de résultats incorrect.

La clause IMMUTABLE n'est actuellement pas prise en charge pour Lambda UDFs.

LAMBDA 'lambda_fn_name'

Nom de la fonction qu'Amazon Redshift appelle.

Pour connaître les étapes de création d'une AWS Lambda fonction, voir Création d'une fonction Lambda avec la console dans le Guide du AWS Lambda développeur.

Pour obtenir des informations sur les autorisations requises pour la fonction Lambda, consultez Autorisations AWS Lambda dans le Guide du développeur AWS Lambda .

IAM_ROLE {par défaut | 'arn:aws:iam : :role/ '*<Compte AWS-id><role-name>*

Utilisez le mot clé par défaut pour qu'Amazon Redshift utilise le rôle IAM défini par défaut et associé au cluster lorsque la commande CREATE EXTERNAL FUNCTION s'exécute.

Utilisez l'Amazon Resource Name (ARN) d'un rôle IAM que votre cluster utilise pour l'authentification et l'autorisation. La commande CREATE EXTERNAL FUNCTION est autorisée à invoquer les fonctions Lambda via ce rôle IAM. Si votre cluster dispose d'un rôle IAM existant avec les autorisations pour invoquer les fonctions Lambda attachées, vous pouvez remplacer l'ARN de votre rôle. Pour plus d'informations, consultez Configuration du paramètre d'autorisation pour Lambda UDFs.

L'exemple suivant montre la syntaxe du paramètre IAM_ROLE.

```
IAM_ROLE 'arn:aws:iam::aws-account-id:role/role-name'
```

RETRY_TIMEOUT millisecondes

Durée totale en millisecondes utilisée par Amazon Redshift pour les retards dans les interruptions de nouvelles tentatives.

Au lieu de lancer de nouvelles tentatives immédiatement pour toute requête ayant échoué, Amazon Redshift effectue des interruptions et attend un certain temps entre les nouvelles tentatives. Amazon Redshift lance ensuite la nouvelle demande pour exécuter à nouveau la requête ayant échoué jusqu'à ce que la somme de tous les retards soit égale ou supérieure à la valeur RETRY_TIMEOUT que vous avez spécifiée. La valeur par défaut est de 20 000 millisecondes.

Lorsqu'une fonction Lambda est appelée, Amazon Redshift lance une nouvelle tentative pour les requêtes qui reçoivent des erreurs telles que `TooManyRequestsException`, `EC2ThrottledException` et `ServiceException`.

Vous pouvez définir le paramètre RETRY_TIMEOUT sur 0 millisecondes pour empêcher toute nouvelle tentative pour une UDF Lambda.

Nombre de MAX_BATCH_ROWS

Nombre maximum de lignes qu'Amazon Redshift envoie dans une seule demande par lot pour une seule invocation Lambda.

La valeur minimale pour ce paramètre est 1. La valeur maximale est INT_MAX ou 2 147 483 647.

Ce paramètre est facultatif. La valeur maximale est INT_MAX ou 2 147 483 647.

Taille MAX_BATCH_SIZE [KB | MB]

Taille maximum de la charge utile de données qu'Amazon Redshift envoie dans une seule demande par lot pour une seule invocation Lambda.

La valeur minimale pour ce paramètre est 1 Ko. La valeur maximale est 5 Mo.

La valeur par défaut de ce paramètre est 5 Mo.

Ko et Mo sont facultatifs. Si vous ne définissez pas l'unité de mesure, Amazon Redshift utilise Ko par défaut.

Notes d'utilisation

Tenez compte des points suivants lorsque vous créez Lambda UDFs :

- L'ordre des appels de fonctions Lambda au niveau des arguments d'entrée n'est ni fixe ni garanti. Il peut varier selon les instances de requêtes en cours d'exécution, en fonction de la configuration du cluster.

- Il n’est pas garanti que les fonctions soient appliquées une seule fois à chaque argument d’entrée. L'interaction entre Amazon Redshift et Amazon Redshift AWS Lambda peut entraîner des appels répétitifs avec les mêmes entrées.

Exemples

Vous trouverez ci-dessous des exemples d'utilisation de fonctions scalaires Lambda définies par l'utilisateur (). UDFs

Exemple de fonction scalaire UDF Lambda utilisant une fonction Node.js Lambda

L’exemple suivant crée une fonction externe appelée `exfunc_sum` qui prend deux entiers comme arguments d’entrée. Cette fonction renvoie la somme sous la forme d’une sortie de nombre entier. Le nom de la fonction Lambda à appeler est `lambda_sum`. Le langage utilisé pour cette fonction Lambda est Node.js 12.x. Veillez à spécifier le rôle IAM. L’exemple utilise `'arn:aws:iam::123456789012:user/johndoe'` en tant que rôle IAM.

```
CREATE EXTERNAL FUNCTION exfunc_sum(INT,INT)
RETURNS INT
VOLATILE
LAMBDA 'lambda_sum'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test';
```

La fonction Lambda prend la charge utile de la requête et effectue une itération sur chaque ligne. Toutes les valeurs d’une seule ligne sont ajoutées pour calculer la somme de cette ligne, qui est enregistrée dans la table de réponses. Le nombre de lignes dans la table de résultats est similaire au nombre de lignes reçues dans la charge utile de la requête.

La charge utile de la réponse JSON doit avoir les données de résultat dans le champ « results » pour être reconnue par la fonction externe. Le champ arguments de la requête envoyée à la fonction Lambda contient la charge utile de données. Il peut y avoir plusieurs lignes dans la charge utile de données en cas de requête par lots. La fonction Lambda suivante effectue une itération sur toutes les lignes de la charge utile de données de la requête. Elle effectue également une itération individuelle sur toutes les valeurs d’une seule ligne.

```
exports.handler = async (event) => {
    // The 'arguments' field in the request sent to the Lambda function contains the
 data payload.
    var t1 = event['arguments'];
```

```
    // 'len(t1)' represents the number of rows in the request payload.
    // The number of results in the response payload should be the same as the number
 of rows received.
    const resp = new Array(t1.length);

    // Iterating over all the rows in the request payload.
    for (const [i, x] of t1.entries())
    {
        var sum = 0;
        // Iterating over all the values in a single row.
        for (const y of x) {
            sum = sum + y;
        }
        resp[i] = sum;
    }
    // The 'results' field should contain the results of the lambda call.
    const response = {
        results: resp
    };
    return JSON.stringify(response);
};
```

L'exemple suivant appelle la fonction externe avec des valeurs littérales.

```
select exfunc_sum(1,2);
exfunc_sum
------------
 3
(1 row)
```

L'exemple suivant crée une table appelée t_sum avec deux colonnes, c1 et c2, du type de données entier et insère deux lignes de données. Ensuite, la fonction externe est appelée en transmettant les noms de colonne de cette table. Les deux lignes de la table sont envoyées dans une requête par lots dans la charge utile de la requête sous la forme d'une seule invocation Lambda.

```
CREATE TABLE t_sum(c1 int, c2 int);
INSERT INTO t_sum VALUES (4,5), (6,7);
SELECT exfunc_sum(c1,c2) FROM t_sum;
 exfunc_sum
---------------
 9
 13
```

```
(2 rows)
```

Exemple de fonction scalaire UDF Lambda utilisant l'attribut RETRY_TIMEOUT

Dans la section suivante, vous trouverez un exemple d'utilisation de l'attribut RETRY_TIMEOUT dans Lambda. UDFs

AWS Lambda les fonctions ont des limites de simultanéité que vous pouvez définir pour chaque fonction. Pour plus d'informations sur les limites de simultanéité, consultez la section Gestion de la simultanéité pour une fonction Lambda dans le Guide du AWS Lambda développeur et l'article Gestion de la simultanéité des AWS Lambda fonctions sur le blog Compute. AWS

Lorsque le nombre de requêtes traitées par une fonction UDF Lambda dépasse les limites de simultanéité, les nouvelles requêtes renvoient l'erreur `TooManyRequestsException`. La fonction UDF Lambda lance une nouvelle tentative pour cette erreur jusqu'à ce que la somme de tous les retards entre les requêtes envoyées à la fonction Lambda soit égale ou supérieure à la valeur RETRY_TIMEOUT que vous avez définie. La valeur RETRY_TIMEOUT par défaut est de 20 000 millisecondes.

L'exemple suivant utilise une fonction Lambda nommée `exfunc_sleep_3`. Cette fonction prend la charge utile de la requête, effectue une itération sur chaque ligne et convertit l'entrée en majuscules. Ensuite, elle passe en veille pendant 3 secondes puis renvoie le résultat. Le langage utilisé pour cette fonction Lambda est Python 3.8.

Le nombre de lignes dans la table de résultats est similaire au nombre de lignes reçues dans la charge utile de la requête. La charge utile de la réponse JSON doit avoir les données de résultat dans le champ `results` pour être reconnue par la fonction externe. Le champ `arguments` dans la requête envoyée à la fonction Lambda contient la charge utile de données. Dans le cas d'une requête par lots, plusieurs lignes peuvent apparaître dans la charge utile de données.

La limite de simultanéité pour cette fonction est spécifiquement définie sur 1 dans la simultanéité réservée pour démontrer l'utilisation de l'attribut RETRY_TIMEOUT. Lorsque l'attribut est défini sur 1, la fonction Lambda ne peut servir qu'une seule requête à la fois.

```
import json
import time
def lambda_handler(event, context):
    t1 = event['arguments']
    # 'len(t1)' represents the number of rows in the request payload.
    # The number of results in the response payload should be the same as the number of
 rows received.
```

```
    resp = [None]*len(t1)

    # Iterating over all rows in the request payload.
    for i, x in enumerate(t1):
        # Iterating over all the values in a single row.
        for j, y in enumerate(x):
            resp[i] = y.upper()

    time.sleep(3)
    ret = dict()
    ret['results'] = resp
    ret_json = json.dumps(ret)
    return ret_json
```

Voici deux exemples supplémentaires illustrant l'attribut RETRY_TIMEOUT. Ils invoquent chacun une seule fonction UDF Lambda. Lorsque la fonction UDF Lambda est appelée, chaque exemple exécute la même requête SQL pour appeler la fonction UDF Lambda à partir de deux séances de base de données simultanées. Lorsque la première requête qui appelle la fonction UDF Lambda est servie par la fonction UDF, la deuxième requête renvoie l'erreur `TooManyRequestsException`. Cela se produit parce que vous avez spécifiquement défini la simultanéité réservée dans la fonction UDF sur 1. Pour obtenir des informations sur la définition de la simultanéité réservée pour les fonctions Lambda, consultez Configuration de la simultanéité réservée.

Le premier exemple ci-dessous définit l'attribut RETRY_TIMEOUT de la fonction UDF Lambda sur 0 milliseconde. Si la requête Lambda reçoit des exceptions de la fonction Lambda, Amazon Redshift n'effectue pas de nouvelle tentative. Ce résultat se produit car l'attribut RETRY_TIMEOUT est défini sur 0.

```
CREATE OR REPLACE EXTERNAL FUNCTION exfunc_upper(varchar)
RETURNS varchar
VOLATILE
LAMBDA 'exfunc_sleep_3'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test'
RETRY_TIMEOUT 0;
```

Lorsque la valeur RETRY_TIMEOUT est définie sur 0, vous pouvez exécuter les deux requêtes suivantes à partir de séances de base de données distinctes pour afficher des résultats différents.

La première requête SQL qui utilise la fonction UDF Lambda s'exécute avec succès.

```
select exfunc_upper('Varchar');
```

```
 exfunc_upper
 --------------
 VARCHAR
(1 row)
```

La deuxième requête, qui est exécutée à partir d'une séance de base de données distincte en même temps, renvoie l'erreur `TooManyRequestsException`.

```
select exfunc_upper('Varchar');
ERROR:  Rate Exceeded.; Exception: TooManyRequestsException; ShouldRetry: 1
DETAIL:
-----------------------------------------------
error:  Rate Exceeded.; Exception: TooManyRequestsException; ShouldRetry: 1
code:      32103
context:query:     0
location:  exfunc_client.cpp:102
process:   padbmaster [pid=26384]
-----------------------------------------------
```

Le deuxième exemple ci-dessous définit l'attribut RETRY_TIMEOUT de la fonction UDF Lambda sur 3 000 millisecondes. Même si la deuxième requête est exécutée simultanément, la fonction UDF Lambda recommence jusqu'à ce que le total des délais atteigne 3 000 millisecondes. Ainsi, les deux requêtes s'exécutent avec succès.

```
CREATE OR REPLACE EXTERNAL FUNCTION exfunc_upper(varchar)
RETURNS varchar
VOLATILE
LAMBDA 'exfunc_sleep_3'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test'
RETRY_TIMEOUT 3000;
```

Lorsque RETRY_TIMEOUT est défini sur 3 000 millisecondes, vous pouvez exécuter les deux requêtes suivantes à partir de séances de base de données distinctes pour afficher les mêmes résultats.

La première requête SQL qui exécute la fonction UDF Lambda s'exécute avec succès.

```
select exfunc_upper('Varchar');
 exfunc_upper
 --------------
 VARCHAR
```

```
(1 row)
```

La deuxième requête s'exécute simultanément, et la fonction UDF Lambda lance de nouvelles tentatives jusqu'à ce que le total des délais atteigne 3 000 millisecondes.

```
select exfunc_upper('Varchar');
 exfunc_upper
--------------
 VARCHAR
(1 row)
```

Exemple de fonction scalaire UDF Lambda utilisant une fonction Python Lambda

L'exemple suivant crée une fonction externe appelée `exfunc_multiplication` qui multiplie les nombres et renvoie un nombre entier. Cet exemple intègre les champs success et `error_msg` dans la réponse Lambda. Le champ success est défini sur false lorsqu'il y a un dépassement d'entier dans le résultat de la multiplication et que la valeur du message `error_msg` est définie sur `Integer multiplication overflow`. La fonction `exfunc_multiplication` prend trois nombres entiers comme arguments d'entrée et renvoie la somme sous la forme d'une sortie de nombre entier.

Le nom de la fonction Lambda qui est appelée est `lambda_multiplication`. Le langage utilisé pour cette fonction Lambda est Python 3.8. Veillez à spécifier le rôle IAM.

```
CREATE EXTERNAL FUNCTION exfunc_multiplication(int, int, int)
RETURNS INT
VOLATILE
LAMBDA 'lambda_multiplication'
IAM_ROLE 'arn:aws:iam::123456789012:role/Redshift-Exfunc-Test';
```

La fonction Lambda prend la charge utile de la requête et effectue une itération sur chaque ligne. Toutes les valeurs d'une seule ligne sont multipliées pour calculer le résultat de cette ligne, qui est enregistré dans la liste de réponses. Cet exemple utilise une valeur success booléenne définie sur true par défaut. Si le résultat de multiplication d'une ligne a un dépassement d'entier, la valeur success est définie sur false. Ensuite, la boucle d'itération s'arrête.

Lorsque la charge utile de la réponse est créée, si la valeur success est false, la fonction Lambda suivante ajoute le champ `error_msg` dans la charge utile. Elle définit également le message d'erreur sur `Integer multiplication overflow`. Si la valeur success est true, les données de résultat sont ajoutées dans le champ results. Le nombre de lignes dans la table de résultats, le cas échéant, est similaire au nombre de lignes reçues dans la charge utile de la requête.

Le champ arguments de la requête envoyée à la fonction Lambda contient la charge utile de données. Il peut y avoir plusieurs lignes dans la charge utile de données en cas de demande par lots. La fonction Lambda suivante effectue une itération sur toutes les lignes de la charge utile de données de la requête et effectue une itération individuelle sur toutes les valeurs au sein d'une seule ligne.

```
import json
def lambda_handler(event, context):
    t1 = event['arguments']
    # 'len(t1)' represents the number of rows in the request payload.
    # The number of results in the response payload should be the same as the number of
 rows received.
    resp = [None]*len(t1)

    # By default success is set to 'True'.
    success = True
    # Iterating over all rows in the request payload.
    for i, x in enumerate(t1):
        mul = 1
        # Iterating over all the values in a single row.
        for j, y in enumerate(x):
            mul = mul*y

        # Check integer overflow.
        if (mul >= 9223372036854775807 or mul <= -9223372036854775808):
            success = False
            break
        else:
            resp[i] = mul
    ret = dict()
    ret['success'] = success
    if not success:
        ret['error_msg'] = "Integer multiplication overflow"
    else:
        ret['results'] = resp
    ret_json = json.dumps(ret)

    return ret_json
```

L'exemple suivant appelle la fonction externe avec des valeurs littérales.

```
SELECT exfunc_multiplication(8, 9, 2);
  exfunc_multiplication
--------------------------
```

```
          144
(1 row)
```

L'exemple suivant crée une table nommée t_multi avec trois colonnes, c1, c2 et c3, du type de données entier. La fonction externe est appelée en transmettant les noms de colonnes de cette table. Les données sont insérées de manière à provoquer un dépassement d'entier afin de montrer comment l'erreur est propagée.

```
CREATE TABLE t_multi (c1 int, c2 int, c3 int);
INSERT INTO t_multi VALUES (2147483647, 2147483647, 4);
SELECT exfunc_multiplication(c1, c2, c3) FROM t_multi;
DETAIL:
  -----------------------------------------------
  error:  Integer multiplication overflow
  code:      32004context:
  context:
  query:     38
  location:  exfunc_data.cpp:276
  process:   query2_16_38 [pid=30494]
  -----------------------------------------------
```

CRÉER UN MODÈLE EXTERNE

Rubriques

- Prérequis pour CREATE EXTERNAL MODEL
- Privilèges requis
- Contrôle des coûts
- Syntaxe CREATE EXTERNAL MODEL
- PARAMÈTRES ET RÉGLAGES DU MODÈLE EXTERNE CREATE
- Paramètres de la fonction d'inférence CREATE EXTERNAL MODEL

Prérequis pour CREATE EXTERNAL MODEL

Avant d'utiliser l'instruction CREATE EXTERNAL MODEL, remplissez les conditions requises dansConfiguration du cluster pour l'utilisation d'Amazon Redshift ML. Vous trouverez ci-dessous un résumé général des étapes requises.

- Créez un cluster Amazon Redshift à l'aide de la console de AWS gestion ou de l'interface de ligne de AWS commande (AWS CLI).
- Attachez la politique AWS Identity and Access Management (IAM) lors de la création du cluster.
- Pour permettre à Amazon Redshift et Amazon Bedrock d'assumer le rôle d'interaction avec d'autres services, ajoutez la politique de confiance appropriée au rôle IAM.
- Activez l'accès aux informations spécifiques LLMs que vous souhaitez utiliser depuis la console Amazon Bedrock.
- (Facultatif) Si vous rencontrez des exceptions de limitation provenant d'Amazon Bedrock, par exemple`Too many requests, please wait before trying again`, même avec de petites quantités de données, vérifiez les quotas dans la section Service Quotas de votre compte Amazon Bedrock. Vérifiez que le quota appliqué au niveau du compte est au moins identique à la valeur de quota AWS par défaut pour les InvokeModeldemandes relatives au modèle que vous utilisez.

Pour plus d’informations sur le rôle IAM, la politique d’approbation et d’autres prérequis, consultez Configuration du cluster pour l'utilisation d'Amazon Redshift ML.

Privilèges requis

Les privilèges requis pour CREATE EXTERNAL MODEL sont les suivants :

- Superuser
- Utilisateurs disposant du privilège CREATE MODEL
- Rôles avec le privilège GRANT CREATE MODEL

Contrôle des coûts

Amazon Redshift ML utilise les ressources existantes du cluster pour créer des modèles de prédiction, de sorte que vous n’avez pas à payer de frais supplémentaires. Toutefois, les AWS frais d'utilisation d'Amazon Bedrock dépendent du modèle que vous sélectionnez. Pour obtenir plus d’informations, consultez Costs for using Amazon Redshift ML (Coûts d’utilisation d’Amazon Redshift ML).

Syntaxe CREATE EXTERNAL MODEL

Voici la syntaxe complète de l'instruction CREATE EXTERNAL MODEL.

```
CREATE EXTERNAL MODEL model_name
FUNCTION function_name
IAM_ROLE {default/'arn:aws:iam::<account-id>:role/<role-name>'}
MODEL_TYPE BEDROCK
SETTINGS (
   MODEL_ID model_id
   [, PROMPT 'prompt prefix']
   [, SUFFIX 'prompt suffix']
   [, REQUEST_TYPE {RAW|UNIFIED}]
   [, RESPONSE_TYPE {VARCHAR|SUPER}]
);
```

La `CREATE EXTERNAL MODEL` commande crée une fonction d'inférence que vous utilisez pour générer du contenu.

Voici la syntaxe d'une fonction d'inférence `CREATE EXTERNAL MODEL` créée à l'aide d'un `REQUEST_TYPE` of `RAW` :

```
SELECT inference_function_name(request_super)
[FROM table];
```

Voici la syntaxe d'une fonction d'inférence `CREATE EXTERNAL MODEL` créée à l'aide d'un `REQUEST_TYPE` of `UNIFIED` :

```
SELECT inference_function_name(input_text, [, inference_config [,
 additional_model_request_fields]])
[FROM table];
```

Pour plus d'informations sur l'utilisation de la fonction d'inférence, consultezUtilisation d'un modèle externe pour l'intégration d'Amazon Redshift ML à Amazon Bedrock.

PARAMÈTRES ET RÉGLAGES DU MODÈLE EXTERNE CREATE

Cette section décrit les paramètres et les réglages de la `CREATE EXTERNAL MODEL` commande.

Rubriques

- PARAMÈTRES CREATE EXTERNAL MODEL
- Paramètres de CREATE EXTERNAL MODEL

PARAMÈTRES CREATE EXTERNAL MODEL

model_name

Nom du modèle externe. Le nom du modèle d'un schéma doit être unique.

FONCTION nom_fonction (type de données [,...])

Nom de la fonction d'inférence qui `CREATE EXTERNAL MODEL` crée. Vous utilisez la fonction d'inférence pour envoyer des demandes à Amazon Bedrock et récupérer du texte généré par ML.

<account-id><role-name>IAM_ROLE {par défaut | 'arn:aws:iam : :role/ '}

Rôle IAM utilisé par Amazon Redshift pour accéder à Amazon Bedrock. Pour obtenir des informations sur le rôle IAM, consultez Création ou mise à jour d'un rôle IAM pour l'intégration d'Amazon Redshift ML à Amazon Bedrock.

MODEL_TYPE SUBSTRATUM

Spécifie le type de modèle. La seule valeur valide est `BEDROCK`.

RÉGLAGES (MODEL_ID model_id [,...])

Spécifie les paramètres du modèle externe. Consultez la section suivante pour plus de détails.

Paramètres de CREATE EXTERNAL MODEL

ID du modèle ID du modèle

L'identifiant du modèle externe, par exemple,`anthropic.claude-v2`. Pour plus d'informations sur le modèle Amazon Bedrock IDs, consultez le modèle Amazon Bedrock. IDs

PROMPT « préfixe rapide »

Spécifie une invite statique qu'Amazon Redshift ajoute au début de chaque demande d'inférence. Compatible uniquement avec un `REQUEST_TYPE` de`UNIFIED`.

SUFFIXE « suffixe rapide »

Spécifie une invite statique qu'Amazon Redshift ajoute à la fin de chaque demande d'inférence. Compatible uniquement avec un `REQUEST_TYPE` de`UNIFIED`.

REQUEST_TYPE {BRUT | UNIFIÉ}

Spécifie le format de la demande envoyée à Amazon Bedrock. Les valeurs valides sont notamment les suivantes :

- RAW : La fonction d'inférence prend l'entrée comme une super valeur unique et renvoie toujours une super valeur. Le format de la super valeur est spécifique au modèle Amazon Bedrock sélectionné. Un super est un modèle de prédiction qui combine plusieurs algorithmes pour produire une seule prédiction améliorée.
- UNIFIÉ : La fonction d'inférence utilise l'API unifiée. Tous les modèles disposent d'une interface unifiée et cohérente avec Amazon Bedrock. Cela fonctionne pour tous les modèles compatibles avec les messages. Cette valeur est celle par défaut.

  Pour plus d'informations, consultez la documentation de l'API Converse dans la documentation de l'API Amazon Bedrock.

RESPONSE_TYPE {VARCHAR | SUPER}

Spécifie le format de la réponse. Si `REQUEST_TYPE` c'est le cas`RAW`, `RESPONSE_TYPE` c'est obligatoire et la seule valeur valide est`SUPER`. Pour toutes les autres `REQUEST TYPE` valeurs, la valeur par défaut `RESPONSE_TYPE` est facultative. `VARCHAR` Les valeurs valides sont notamment les suivantes :

- VARCHAR : Amazon Redshift renvoie uniquement le texte de réponse généré par le modèle.
- SUPER : Amazon Redshift renvoie l'intégralité de la réponse JSON générée par le modèle sous forme de super. Cela inclut la réponse textuelle et des informations telles que le motif de l'arrêt et l'utilisation du jeton d'entrée et de sortie du modèle. Un super est un modèle de prédiction qui combine plusieurs algorithmes pour produire une seule prédiction améliorée.

Paramètres de la fonction d'inférence CREATE EXTERNAL MODEL

Cette section décrit les paramètres valides pour la fonction d'inférence créée par la `CREATE EXTERNAL MODEL` commande.

Paramètres de la fonction d'inférence CREATE EXTERNAL MODEL pour **`REQUEST_TYPERAW`**

Une fonction d'inférence créée avec un `REQUEST_TYPE` de `RAW` possède un super argument d'entrée et renvoie toujours un super type de données. La syntaxe du super d'entrée suit la syntaxe de la demande du modèle spécifique sélectionné sur Amazon Bedrock.

Paramètres de la fonction d'inférence CREATE EXTERNAL MODEL pour **`REQUEST_TYPEUNIFIED`**

input_text

Le texte qu'Amazon Redshift envoie à Amazon Bedrock.

inférence_config

Une super valeur qui contient des paramètres facultatifs qu'Amazon Redshift envoie à Amazon Bedrock. Il peut s'agir notamment des éléments suivants :

- maxTokens
- Séquences d'arrêt
- temperature
- topP

Ces paramètres sont tous facultatifs et distinguent tous les majuscules et minuscules. Pour plus d'informations sur ces paramètres, consultez le InferenceConfigurationmanuel Amazon Bedrock API Reference.

CREATE EXTERNAL SCHEMA

Crée un schéma externe dans la base de données actuelle. Vous pouvez utiliser ce schéma externe pour vous connecter à des bases de données Amazon RDS for PostgreSQL ou d'édition compatible avec Amazon Aurora PostgreSQL. Vous pouvez également créer un schéma externe qui fait référence à une base de données dans un catalogue de données externe tel qu' AWS Glue Athena ou à une base de données dans un métastore Apache Hive, tel qu'Amazon EMR.

Le propriétaire de ce schéma est l'auteur de la commande CREATE EXTERNAL SCHEMA. Pour transférer la propriété d'un schéma externe, utilisez ALTER SCHEMA pour modifier le propriétaire. Utilisez la commande GRANT pour autoriser d'autres utilisateurs ou groupes d'utilisateurs à accéder au schéma.

Vous ne pouvez pas utiliser les commandes GRANT ou REVOKE pour des autorisations concernant une table externe. Vous pouvez en revanche accorder ou révoquer les autorisations pour le schéma externe.

> **Note**
>
> Si vous disposez actuellement de tables externes Redshift Spectrum dans le catalogue de données Amazon Athena, vous pouvez procéder à la migration de votre catalogue de données Athena vers un AWS Glue Data Catalog. Pour utiliser le catalogue de AWS Glue données avec Redshift Spectrum, vous devrez peut-être modifier vos politiques AWS Identity

and Access Management (IAM). Pour plus d'informations, consultez la section Mise à niveau vers le catalogue de AWS Glue données dans le guide de l'utilisateur d'Athena.

Pour afficher les détails relatifs aux schémas externes, interrogez la vue système SVV_EXTERNAL_SCHEMAS.

Syntaxe

La syntaxe suivante décrit la commande CREATE EXTERNAL SCHEMA utilisée pour référencer des données à l'aide d'un catalogue de données externe. Pour plus d'informations, consultez Amazon Redshift Spectrum.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] local_schema_name
FROM [ [ DATA CATALOG ] | HIVE METASTORE | POSTGRES | MYSQL | KINESIS | MSK | REDSHIFT
 | KAFKA ]
[ DATABASE 'database_name' ]
[ SCHEMA 'schema_name' ]
[ REGION 'aws-region' ]
[ IAM_ROLE [ default | 'SESSION' | 'arn:aws:iam::<Compte AWS-id>:role/<role-name>' ] ]
[ AUTHENTICATION [ none | iam | mtls] ]
[ AUTHENTICATION_ARN 'acm-certificate-arn' | SECRET_ARN 'ssm-secret- arn' ]
[ URI ['hive_metastore_uri' [ PORT port_number ] | 'hostname' [ PORT port_number ] |
 'Kafka bootstrap URL'] ]
[ CLUSTER_ARN 'arn:aws:kafka:<region>:<Compte AWS-id>:cluster/msk/<cluster uuid>' ]
[ CATALOG_ROLE [ 'SESSION' | 'catalog-role-arn-string' ] ]
[ CREATE EXTERNAL DATABASE IF NOT EXISTS ]
[ CATALOG_ID 'Amazon Web Services account ID containing Glue or Lake Formation
 database' ]
```

La syntaxe suivante décrit la commande CREATE EXTERNAL SCHEMA utilisée pour référencer des données à l'aide d'une requête fédérée vers RDS POSTGRES ou Aurora PostgreSQL. Vous pouvez également créer un schéma externe qui fait référence à des sources de streaming, telles que Kinesis Data Streams. Pour de plus amples informations, veuillez consulter Interrogation de données avec requête fédérée dans Amazon Redshift.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] local_schema_name
FROM POSTGRES
DATABASE 'federated_database_name' [SCHEMA 'schema_name']
URI 'hostname' [ PORT port_number ]
IAM_ROLE [ default | 'arn:aws:iam::<Compte AWS-id>:role/<role-name>' ]
```

```
SECRET_ARN 'ssm-secret-arn'
```

La syntaxe suivante décrit la commande CREATE EXTERNAL SCHEMA utilisée pour référencer des données à l'aide d'une requête fédérée vers RDS MySQL ou Aurora MySQL. Pour de plus amples informations, veuillez consulter Interrogation de données avec requête fédérée dans Amazon Redshift.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] local_schema_name
FROM MYSQL
DATABASE 'federated_database_name'
URI 'hostname' [ PORT port_number ]
IAM_ROLE [ default | 'arn:aws:iam::<Compte AWS-id>:role/<role-name>' ]
SECRET_ARN 'ssm-secret-arn'
```

La syntaxe suivante décrit la commande CREATE EXTERNAL SCHEMA utilisée pour référencer des données dans un flux Kinesis. Pour de plus amples informations, veuillez consulter Diffusion de l'ingestion vers une vue matérialisée.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] schema_name
FROM KINESIS
IAM_ROLE [ default | 'arn:aws:iam::<Compte AWS-id>:role/<role-name>' ]
```

La syntaxe suivante décrit la commande CREATE EXTERNAL SCHEMA utilisée pour faire référence au cluster Amazon Managed Streaming for Apache Kafka ou Confluent Cloud et aux rubriques à partir desquelles l'ingestion peut être effectuée. Pour vous connecter, vous devez fournir l'URI du courtier. Pour de plus amples informations, veuillez consulter Diffusion de l'ingestion vers une vue matérialisée.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] schema_name
FROM KAFKA
[ IAM_ROLE [ default | 'arn:aws:iam::<Compte AWS-id>:role/<role-name>' ] ]
URI 'Kafka bootstrap URI'
AUTHENTICATION [ none | iam | mtls ]
[ AUTHENTICATION_ARN 'acm-certificate-arn' | SECRET_ARN 'ssm-secret- arn' ];
```

La syntaxe suivante décrit la commande CREATE EXTERNAL SCHEMA utilisée pour référencer des données à l'aide d'une requête entre bases de données.

```
CREATE EXTERNAL SCHEMA local_schema_name
FROM  REDSHIFT
```

```
DATABASE 'redshift_database_name' SCHEMA 'redshift_schema_name'
```

Paramètres

IF NOT EXISTS

Clause indiquant que si le schéma spécifié existe déjà, la commande ne doit apporter aucune modification et renvoyer un message selon lequel le schéma existe, plutôt que de s'arrêter avec une erreur. Puisque cette clause est utile lors de l'écriture de scripts, le script n'échoue pas si CREATE EXTERNAL SCHEMA tente de créer un schéma qui existe déjà.

local_schema_name

Nom du nouveau schéma externe. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

FROM [DATA CATALOG] | HIVE METASTORE | POSTGRES | MYSQL | KINESIS | MSK | REDSHIFT

Mot-clé indiquant où se situe la base de données externe.

DATA CATALOG indique que la base de données externe est définie dans le catalogue de données Athena ou AWS Glue Data Catalog.

Si la base de données externe est définie dans un catalogue de données externe dans une autre région AWS , le paramètre REGION est requis. La valeur par défaut est DATA CATALOG.

HIVE METASTORE indique que la base de données externe est définie dans un metastore Apache Hive. L'URI est obligatoire si HIVE METASTORE est spécifié.

POSTGRES indique que la base de données externe est définie dans RDS PostgreSQL ou Aurora PostgreSQL.

MYSQL indique que la base de données externe est définie dans RDS MySQL ou Aurora MySQL.

KINESIS indique que la source de données est un flux de Kinesis Data Streams.

MSK indique que la source de données est un cluster Amazon MSK provisionné ou sans serveur.

KAFKA indique que la source de données est un cluster Kafka. Vous pouvez utiliser ce mot clé pour Amazon MSK et Confluent Cloud.

FROM REDSHIFT

Mot-clé indiquant que la base de données se trouve dans Amazon Redshift.

DATABASE ’nom_base_de_données_redshift’ SCHEMA ’nom_schéma_redshift’

Nom de la base de données Amazon Redshift.

Le nom_schéma_redshift indique le schéma dans Amazon Redshift. La valeur par défaut de nom_schéma_redshift est `public`.

DATABASE ’nom_base_de_données_fédérée’

Mot-clé qui indique le nom de la base de données externe dans un moteur de base de données PostgreSQL ou MySQL.

[SCHEMA ’schema_name’]

Le nom_schéma indique le schéma dans un moteur de base de données PostgreSQL pris en charge. Le nom_schéma par défaut est `public`.

Vous ne pouvez pas spécifier de SCHEMA lorsque vous configurez une requête fédérée sur un moteur de base de données MySQL pris en charge.

REGION ’aws-region’

Si la base de données externe est définie dans un catalogue de données Athena ou dans la AWS Glue Data Catalog AWS région dans laquelle se trouve la base de données. Si la base de données est définie dans un catalogue de données externe, ce paramètre est obligatoire.

URI ['hive_metastore_uri' [numéro de port] | « nom d'hôte » [numéro de port] | « URI Kafka bootstrap »]

URI du nom_hôte et numéro_port d’un moteur de base de données PostgreSQL ou MySQL pris en charge. nom_hôte est le nœud principal du jeu de réplicas. Le point de terminaison doit être accessible (routable) à partir du cluster Amazon Redshift. Le numéro de port (port_number) par défaut pour PostgreSQL est 5432. Le numéro de port (port_number) par défaut pour MySQL est 3306.

> **Note**
>
> Le moteur de base de données PostgreSQL ou MySQL pris en charge doit se trouver dans le même VPC que votre cluster Amazon Redshift avec un groupe de sécurité liant Amazon Redshift et RDS URL-RSPostgreSQL ou Aurora PostgreSQL. En outre, vous pouvez utiliser le routage VPC amélioré pour configurer un cas d'utilisation inter-VPC. Pour plus d'informations, consultez la section Points de terminaison VPC gérés par Redshift.

Spécification d'un URI de métastore Hive

Si la base de données se trouve dans un metastore Hive, spécifiez l'URI et éventuellement le numéro de port du metastore. Le numéro de port par défaut est 9083.

Un URI ne contient pas de spécification de protocole (« http:// »). Voici un exemple d'URI valide : `uri '172.10.10.10'`.

Spécification d'un URI de courtier pour l'ingestion du streaming

L'inclusion de l'URI bootstrap-broker permet de se connecter à un cluster Amazon MSK ou Confluent Cloud et de recevoir des données en streaming. Pour plus d'informations et pour voir un exemple, consultez Getting started with streaming ingestion from Amazon Managed Streaming for Apache Kafka.

IAM_ROLE [par défaut | 'SESSION' | 'arn:aws:iam : :role/ '] *<Compte AWS-id> <role-name>*

Utilisez le mot clé par défaut pour qu'Amazon Redshift utilise le rôle IAM défini par défaut et associé au cluster lorsque la commande CREATE EXTERNAL SCHEMA s'exécute.

Utilisez `'SESSION'` si vous vous connectez à votre cluster Amazon Redshift à l'aide d'une identité fédérée et que vous accédez aux tables à partir du schéma externe créé à l'aide de cette commande. Pour plus d'informations, consultez Utilisation d'une identité fédérée pour gérer l'accès à Amazon Redshift aux ressources locales et aux tables externes Amazon Redshift Spectrum, qui explique comment configurer l'identité fédérée. Notez que cette configuration, qui utilise `'SESSION'` à la place de l'ARN, ne peut être utilisée que si le schéma est créé à l'aide de `DATA CATALOG`.

Utilisez l'Amazon Resource Name (ARN) d'un rôle IAM que votre cluster utilise pour l'authentification et l'autorisation. Au minimum, le rôle IAM doit être autorisé à exécuter une opération LIST sur le compartiment Amazon S3 devant être accessible et une opération GET sur les objets Amazon S3 contenus dans le compartiment.

Le code suivant montre la syntaxe de la chaîne de paramètre IAM_ROLE pour un seul ARN.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Vous pouvez créer des chaînes de rôles pour permettre à votre cluster d'endosser un autre rôle IAM, y compris un rôle appartenant à un autre compte. Les chaînes ainsi créées peuvent inclure jusqu'à 10 rôles. Pour voir un exemple de création de chaîne de rôles, consultez Créer des rôles IAM dans Amazon Redshift Spectrum.

Attachez à ce rôle IAM une politique d'autorisations IAM similaire à la suivante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-west-2:123456789012:secret:my-
rds-secret-VNenFy"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour connaître les étapes à suivre afin de créer un rôle IAM à utiliser avec une requête fédérée, consultez Création d'un secret et d'un rôle IAM pour utiliser des requêtes fédérées.

> **Note**
>
> N'incluez pas d'espaces dans la liste des rôles chaînés.

L'exemple suivant montre la syntaxe d'une chaîne de trois rôles.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-1-name>,arn:aws:iam::<aws-
account-id>:role/<role-2-name>,arn:aws:iam::<aws-account-id>:role/<role-3-name>'
```

SECRET_ARN « » ssm-secret-arn

Le nom de ressource Amazon (ARN) d'un secret de moteur de base de données PostgreSQL ou MySQL pris en charge créé à l'aide de. AWS Secrets Manager Pour plus d'informations sur la création et la récupération d'un ARN pour un secret, consultez les sections Gérer les secrets AWS Secrets Manager dans le guide de AWS Secrets Manager l'utilisateur et Extraction du nom de ressource Amazon (ARN) du secret dans Amazon Redshift.

CATALOG_ROLE ['SESSION' |] catalog-role-arn-string

`'SESSION'` À utiliser pour vous connecter à votre cluster Amazon Redshift à l'aide d'une identité fédérée à des fins d'authentification et d'autorisation du catalogue de données. Pour plus d'informations sur la réalisation des étapes relatives à l'identité fédérée, consultez Utilisation d'une identité fédérée pour gérer l'accès d'Amazon Redshift aux ressources locales et aux tables externes Amazon Redshift Spectrum. Notez que le `'SESSION'` rôle ne peut être utilisé que si le schéma est créé dans DATA CATALOG.

Nom Amazon Resource Name (ARN) d'un rôle IAM que votre cluster utilise pour l'authentification et l'autorisation.

Si CATALOG_ROLE n'est pas spécifié, Amazon Redshift utilise la valeur IAM_ROLE spécifiée. Le rôle du catalogue doit être autorisé à accéder au catalogue de données dans AWS Glue ou Athena. Pour de plus amples informations, veuillez consulter Politiques IAM pour Amazon Redshift Spectrum.

Le code suivant montre la syntaxe de la chaîne de paramètre CATALOG_ROLE pour un seul ARN.

```
CATALOG_ROLE 'arn:aws:iam::<aws-account-id>:role/<catalog-role>'
```

Vous pouvez créer des chaînes de rôles pour permettre à votre cluster d'endosser un autre rôle IAM, y compris un rôle appartenant à un autre compte. Les chaînes ainsi créées peuvent inclure jusqu'à 10 rôles. Pour plus d'informations, consultez Créer des rôles IAM dans Amazon Redshift Spectrum.

> **Note**
>
> La liste des rôles de la chaîne ne doit pas inclure d'espaces.

L'exemple suivant montre la syntaxe d'une chaîne de trois rôles.

```
CATALOG_ROLE 'arn:aws:iam::<aws-account-id>:role/<catalog-role-1-name>,arn:aws:iam::<aws-account-id>:role/<catalog-role-2-name>,arn:aws:iam::<aws-account-id>:role/<catalog-role-3-name>'
```

CREATE EXTERNAL DATABASE IF NOT EXISTS

Clause qui crée une base de données externe avec le nom spécifié par l'argument DATABASE, si la base de données externe spécifiée n'existe pas. La commande n'apporte aucune modification si la base de données externe spécifiée existe. Dans ce cas, la commande renvoie un message indiquant que la base de données externe existe, plutôt que de s'arrêter avec une erreur.

> **Note**
>
> La clause CREATE EXTERNAL DATABASE IF NOT EXISTS ne peut pas être utilisée avec HIVE METASTORE.
> Pour utiliser CREATE EXTERNAL DATABASE IF NOT EXISTS avec un catalogue de données activé pour AWS Lake Formation, vous devez disposer de l'autorisation `CREATE_DATABASE` sur le catalogue de données.

CATALOG_ID « ID de compte Amazon Web Services contenant la base de données Glue ou Lake Formation »

L'identifiant du compte sur lequel la base de données du catalogue de données est stockée.

`CATALOG_ID`peut être spécifiée uniquement si vous prévoyez de vous connecter à votre cluster Amazon Redshift ou à Amazon Redshift sans serveur à l'aide d'une identité fédérée pour l'authentification et l'autorisation du catalogue de données en définissant l'une des options suivantes :

- `CATALOG_ROLE` sur `'SESSION'`
- `IAM_ROLE`à`'SESSION'` et`'CATALOG_ROLE'` régler sur sa valeur par défaut

Pour plus d'informations sur la réalisation des étapes relatives à l'identité fédérée, consultez Utilisation d'une identité fédérée pour gérer l'accès d'Amazon Redshift aux ressources locales et aux tables externes Amazon Redshift Spectrum.

AUTHENTICATION

Type d’authentification défini pour l’ingestion en streaming. L’ingestion en streaming assortie de types d’authentification fonctionne avec Amazon Managed Streaming for Apache Kafka. Les types `AUTHENTICATION` sont les suivants :

- none — Spécifie qu'aucune authentification n'est requise. Cela correspond à un accès non authentifié sur MSK ou en texte brut avec TLS sur Apache Kafka.
- iam – Indique une authentification IAM. Lorsque vous choisissez cette option, vérifiez que le rôle IAM dispose des autorisations nécessaires à l’authentification IAM. Pour en savoir plus sur la définition du schéma externe, consultez Commencer à ingérer du streaming à partir de sources Apache Kafka.
- mtls — Spécifie que la sécurité mutuelle de la couche de transport assure une communication sécurisée en facilitant l'authentification entre un client et un serveur. Dans ce cas, le client est Redshift et le serveur est Amazon MSK. Pour plus d'informations sur la configuration de l'ingestion du streaming avec les mTLS, consultezAuthentification avec MTL pour l'ingestion de flux Redshift à partir de sources Apache Kafka.

AUTHENTIFICATION_ARN

L'ARN du AWS Certificate Manager certificat utilisé par Amazon Redshift pour l'authentification MTLS avec Amazon MSK. L'ARN est disponible dans la console ACM lorsque vous choisissez le certificat émis.

CLUSTER_ARN

Pour l'ingestion du streaming, le CLUSTER_ARN est l'identifiant du cluster Amazon Managed Streaming for Apache Kafka à partir duquel vous diffusez. Lorsque vous utilisez CLUSTER_ARN, une politique de rôle IAM incluant l'autorisation est requise. `kafka:GetBootstrapBrokers` Cette option est fournie à des fins de rétrocompatibilité. Actuellement, nous recommandons d'utiliser l'option d'URI bootstrap-broker pour vous connecter aux clusters Amazon Managed Streaming for Apache Kafka. Pour plus d’informations, consultez Ingestion en streaming (version préliminaire).

Notes d’utilisation

Pour connaître les restrictions relatives à l’utilisation du catalogue de données Athena, consultez Athena Limits dans Références générales AWS.

Pour connaître les limites d'utilisation du AWS Glue Data Catalog, voir AWS Glue Limites dans le Références générales AWS.

Ces restrictions ne s'appliquent pas à un metastore Hive.

Il y a un maximum de 9 900 schémas par base de données. Pour plus d'informations, consultez Quotas et limites dans le Guide de gestion Amazon Redshift.

Pour annuler l'enregistrement du schéma, utilisez la commande DROP SCHEMA.

Pour afficher les détails relatifs aux schémas externes, interrogez les vues système suivantes :

- SVV_EXTERNAL_SCHEMAS
- SVV_EXTERNAL_TABLES
- SVV_EXTERNAL_COLUMNS

Exemples

L'exemple suivant permet de créer un schéma externe en utilisant une base de données dans un catalogue de données nommé `sampledb` dans la région USA Ouest (Oregon). Utilisez cet exemple avec un catalogue de données Athena ou AWS Glue .

```
create external schema spectrum_schema
from data catalog
database 'sampledb'
region 'us-west-2'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole';
```

L'exemple suivant permet de créer un schéma externe ainsi qu'une base de données externe nommée `spectrum_db`.

```
create external schema spectrum_schema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole'
create external database if not exists;
```

L'exemple suivant permet de créer un schéma externe en utilisant une base de données de metastore Hive nommée `hive_db`.

```
create external schema hive_schema
from hive metastore
database 'hive_db'
uri '172.10.10.10' port 99
iam_role 'arn:aws:iam::123456789012:role/MySpectrumRole';
```

L'exemple suivant établit une chaîne de rôles afin d'utiliser le rôle `myS3Role` pour accéder à Amazon S3 et `myAthenaRole` pour l'accès au catalogue de données. Pour plus d'informations, consultez Créer des rôles IAM dans Amazon Redshift Spectrum.

```
create external schema spectrum_schema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myRedshiftRole,arn:aws:iam::123456789012:role/
myS3Role'
catalog_role 'arn:aws:iam::123456789012:role/myAthenaRole'
create external database if not exists;
```

L'exemple suivant crée un schéma externe qui référence une base de données Aurora PostgreSQL.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] myRedshiftSchema
FROM POSTGRES
DATABASE 'my_aurora_db' SCHEMA 'my_aurora_schema'
URI 'endpoint to aurora hostname' PORT 5432
IAM_ROLE 'arn:aws:iam::123456789012:role/MyAuroraRole'
SECRET_ARN 'arn:aws:secretsmanager:us-east-2:123456789012:secret:development/
MyTestDatabase-AbCdEf'
```

L'exemple suivant crée un schéma externe pour renvoyer à la base de données sales_db importée sur le cluster consommateur.

```
CREATE EXTERNAL SCHEMA sales_schema FROM REDSHIFT DATABASE 'sales_db' SCHEMA 'public';
```

L'exemple suivant crée un schéma externe qui référence une base de données Aurora MySQL.

```
CREATE EXTERNAL SCHEMA [IF NOT EXISTS] myRedshiftSchema
FROM MYSQL
DATABASE 'my_aurora_db'
URI 'endpoint to aurora hostname'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyAuroraRole'
```

```
SECRET_ARN 'arn:aws:secretsmanager:us-east-2:123456789012:secret:development/
MyTestDatabase-AbCdEf'
```

CREATE EXTERNAL TABLE

Crée une table externe dans le schéma spécifié. Toutes les tables externes doivent être créées dans un schéma externe. Le chemin de recherche n'est pas pris en charge pour les schémas et tables externes. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL SCHEMA.

Outre les tables externes créées à l'aide de la commande CREATE EXTERNAL TABLE, Amazon Redshift peut faire référence à des tables externes définies dans un AWS Lake Formation catalogue AWS Glue ou un métastore Apache Hive. Utilisez la commande CREATE EXTERNAL SCHEMA pour enregistrer une base de données externe définie dans un catalogue de données externe, et faites en sorte que les tables externes puissent être utilisées dans Amazon Redshift. Si la table externe existe dans un AWS Lake Formation catalogue AWS Glue ou un métastore Hive, vous n'avez pas besoin de créer la table à l'aide de CREATE EXTERNAL TABLE. Pour afficher les tables externes, interrogez la vue système SVV_EXTERNAL_TABLES.

En exécutant la commande CREATE EXTERNAL TABLE AS, vous pouvez créer une table externe basée sur la définition de colonne d'une requête et écrire les résultats de cette requête dans Amazon S3. Les résultats sont au format Apache Parquet ou au format texte délimité. Si la table externe possède une ou plusieurs clés de partition, Amazon Redshift partitionne les nouveaux fichiers en fonction de ces clés de partition et enregistre automatiquement les nouvelles partitions dans le catalogue externe. Pour plus d'informations sur la commande CREATE EXTERNAL TABLE AS, consultez Notes d'utilisation.

Vous pouvez interroger une table externe en utilisant la même syntaxe SELECT que celle que vous utilisez avec d'autres tables Amazon Redshift. Vous pouvez également utiliser la syntaxe INSERT pour écrire de nouveaux fichiers à l'emplacement de la table externe sur Amazon S3. Pour plus d'informations, consultez INSERT (table externe).

Pour créer une vue avec une table externe, incluez la clause WITH NO SCHEMA BINDING dans l'instruction CREATE VIEW.

Vous ne pouvez pas exécuter CREATE EXTERNAL TABLE à l'intérieur d'une transaction (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

Privilèges requis

Pour créer des tables externes, vous devez être le propriétaire du schéma externe ou un superutilisateur. Pour transférer la propriété d'un schéma externe, utilisez ALTER SCHEMA pour modifier le propriétaire. L'accès aux tables externes est contrôlé par l'accès au schéma externe. Vous ne pouvez pas accorder GRANT ni révoquer REVOKE des autorisations pour une table externe. A la place, accordez ou révoquez USAGE sur le schéma externe.

Vous trouverez dans les Notes d'utilisation des informations complémentaires sur les autorisations spécifiques des tables externes.

Syntaxe

```
CREATE EXTERNAL TABLE
external_schema.table_name
(column_name data_type [, …] )
[ PARTITIONED BY (col_name data_type [, … ] )]
[ { ROW FORMAT DELIMITED row_format |
  ROW FORMAT SERDE 'serde_name'
  [ WITH SERDEPROPERTIES ( 'property_name' = 'property_value' [, ...] ) ] } ]
STORED AS file_format
LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file' }
[ TABLE PROPERTIES ( 'property_name'='property_value' [, ...] ) ]
```

Voici la syntaxe de la commande CREATE EXTERNAL TABLE AS.

```
CREATE EXTERNAL TABLE
external_schema.table_name
[ PARTITIONED BY (col_name [, … ] ) ]
[ ROW FORMAT DELIMITED row_format ]
STORED AS file_format
LOCATION { 's3://bucket/folder/' }
[ TABLE PROPERTIES ( 'property_name'='property_value' [, ...] ) ]
 AS
 { select_statement }
```

Paramètres

schéma_externe.nom_table

Nom de la table à créer, qualifié par un nom de schéma externe. Les tables externes doivent être créées dans un schéma externe. Pour plus d'informations, consultez CREATE EXTERNAL SCHEMA.

La longueur maximale d'un nom de table est de 127 octets ; les noms plus longs sont tronqués à 127 octets. Vous pouvez utiliser des caractères multioctets UTF-8 jusqu'à un maximum de quatre octets. Amazon Redshift impose une limite de 9 900 tables par cluster, y compris les tables temporaires définies par l'utilisateur et les tables temporaires créées par Amazon Redshift lors du traitement des requêtes ou de la maintenance du système. Le cas échéant, le nom de la table peut être qualifié avec le nom de la base de données. Dans l'exemple suivant, le nom de base de données est `spectrum_db`, le nom du schéma externe est `spectrum_schema` et le nom de la table est `test`.

```
create external table spectrum_db.spectrum_schema.test (c1 int)
stored as parquet
location 's3://amzn-s3-demo-bucket/myfolder/';
```

Si la base de données ou le schéma spécifié n'existe pas, la table n'est pas créée et l'instruction renvoie une erreur. Vous ne pouvez pas créer de tables ni de vues dans les bases de données système `template0`, `template1`, `padb_harvest` ou `sys:internal`.

Le nom de la table doit être un nom unique pour le schéma spécifié.

Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

(nom_colonne type_données)

Nom et type de données de chaque colonne en cours de création.

La longueur maximale d'un nom de colonne est de 127 octets ; les noms plus longs sont tronqués à 127 octets. Vous pouvez utiliser des caractères multioctets UTF-8 jusqu'à un maximum de quatre octets. Vous ne pouvez pas spécifier de noms de colonne `"$path"` ou `"$size"`. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

Par défaut, Amazon Redshift crée les tables externes avec les pseudo-colonnes `$path` et `$size`. Vous pouvez désactiver la création de pseudo-colonnes d'une séance en définissant le paramètre

de configuration `spectrum_enable_pseudo_columns` avec la valeur `false`. Pour plus d'informations, consultez Pseudocolonnes .

Si les pseudo-colonnes sont activées, le nombre maximal de colonnes que vous pouvez définir dans une seule table est 1 598. Si les pseudo-colonnes ne sont pas activées, le nombre maximal de colonnes que vous pouvez définir dans une seule table est 1 600.

Si vous créez une grande table, assurez-vous que la liste de colonnes ne dépasse pas les limites de largeur de ligne pour les résultats intermédiaires pendant le traitement des charges et des requêtes. Pour plus d'informations, consultez Notes d’utilisation.

Pour une commande CREATE EXTERNAL TABLE AS, aucune liste de colonnes n’est requise, car les colonnes sont dérivées de la requête.

data_type

Les encodages Types de données suivants sont pris en charge :

- PETITE MENTHE () INT2
- ENTIER (INT, INT4)
- BIGINT () INT8
- DECIMAL (NUMERIC)
- RÉEL (FLOAT4)
- DOUBLE PRÉCISION (FLOAT8)
- BOOLEAN (BOOL)
- CHAR (CHARACTER)
- VARCHAR (CHARACTER VARYING)
- VARBYTE (CHARACTER VARYING) : peut être utilisé avec des fichiers de données Parquet et ORC, et uniquement avec des tables non partitionnées.
- DATE : peut être utilisé uniquement avec du texte ou des fichiers de données Parquet ou ORC, ou comme colonne de partition.
- TIMESTAMP

Pour DATE, vous pouvez utiliser les formats décrits ci-dessous. Pour les valeurs de mois représentées à l’aide de chiffres, les formats suivants sont pris en charge :

- `mm-dd-yyyy` Par exemple, `05-01-2017`. Il s’agit de l’option par défaut.

- yyyy-mm-dd, où l'année est représentée par plus de deux chiffres. Par exemple, 2017-05-01.

Pour les valeurs de mois représentées à l'aide de l'abréviation de trois lettres, les formats suivants sont pris en charge :

- mmm-dd-yyyy Par exemple, may-01-2017. Il s'agit de l'option par défaut.
- dd-mmm-yyyy, où l'année est représentée par plus de deux chiffres. Par exemple, 01-may-2017.
- yyyy-mmm-dd, où l'année est représentée par plus de deux chiffres. Par exemple, 2017-may-01.

Pour les valeurs d'années qui sont constamment inférieures à 100, l'année est calculée de la manière suivante :

- Si l'année est inférieure à 70, elle est calculée comme l'année plus 2000. Par exemple, la date 05-01-17 au format mm-dd-yyyy est convertie au format 05-01-2017.
- Si l'année est inférieure à 100 et supérieure à 69, l'année est calculée comme l'année plus 1900. Par exemple, la date 05-01-89 au format mm-dd-yyyy est convertie au format 05-01-1989.
- Pour les valeurs d'année représentées par deux chiffres, ajoutez les zéros de tête pour représenter l'année en quatre chiffres.

Les valeurs d'horodatage dans les fichiers texte doivent être au format yyyy-mm-dd HH:mm:ss.SSSSSS, comme illustré par la valeur d'horodatage suivante : 2017-05-01 11:30:59.000000.

La longueur d'une colonne VARCHAR est définie en octets et non pas en caractères. Par exemple, une colonne VARCHAR(12) peut contenir 12 caractères codés sur un octet ou 6 caractères codés sur deux octets. Lorsque vous interrogez une table externe, les résultats sont tronqués pour s'adapter à la taille définie de la colonne sans renvoyer d'erreur. Pour plus d'informations, consultez Stockage et plages.

Pour de meilleures performances, nous vous recommandons de spécifier la plus petite taille de colonne adaptée à vos données. Pour rechercher la taille maximale en octets des valeurs d'une colonne, utilisez la fonction OCTET_LENGTH. L'exemple suivant renvoie la taille maximale des valeurs de la colonne EMAIL.

```
select max(octet_length(email)) from users;
```

```
max
---
 62
```

PARTITIONED BY (nom_col type_données [, …])

Clause qui définit une table partitionnée avec une ou plusieurs colonnes de partition. Un répertoire de données distinct est utilisé pour chaque combinaison spécifiée, ce qui dans certains cas améliore la performance des requêtes. Les colonnes partitionnées n'existent pas au sein même des données de la table. Si la valeur de nom_col est identique à celle d'une colonne de table, une erreur est renvoyée.

Après avoir créé une table partitionnée, modifiez-la à l'aide d'une instruction ALTER TABLE ... ADD PARTITION pour enregistrer de nouvelles partitions dans le catalogue externe. Lorsque vous ajoutez une partition, vous définissez l'emplacement du sous-dossier sur Amazon S3 qui contient les données de partition.

Par exemple, si la table `spectrum.lineitem_part` est définie avec `PARTITIONED BY (l_shipdate date)`, exécutez la commande ALTER TABLE suivante pour ajouter une partition.

```
ALTER TABLE spectrum.lineitem_part ADD PARTITION (l_shipdate='1992-01-29')
LOCATION 's3://spectrum-public/lineitem_partition/l_shipdate=1992-01-29';
```

Si vous utilisez la commande CREATE EXTERNAL TABLE AS, vous n'avez pas besoin d'exécuter la commande ALTER TABLE...ADD PARTITION. Amazon Redshift enregistre automatiquement les nouvelles partitions dans le catalogue externe. Amazon Redshift écrit également automatiquement les données correspondantes dans les partitions d'Amazon S3 en fonction de la ou des clés de partition définies dans la table.

Pour afficher les partitions, interrogez la vue système SVV_EXTERNAL_PARTITIONS.

> **Note**
>
> Pour une commande CREATE EXTERNAL TABLE AS, vous n'avez pas besoin de spécifier le type de données de la colonne de partition car cette colonne est dérivée de la requête.

ROW FORMAT DELIMITED format_ligne

Clause qui spécifie le format des données sous-jacentes. Les valeurs possibles pour format_ligne sont les suivantes :

- LINES TERMINATED BY ’délimiteur’
- FIELDS TERMINATED BY ’délimiteur’

Spécifiez un caractère ASCII unique pour ’délimiteur’. Vous pouvez spécifier des caractères ASCII non imprimables en octal, au format `'\ddd'`, où *d* est un chiffre octal (de 0 à 7) jusqu’à ‘\177’. L’exemple suivant spécifie le caractère BEL (bell) en octal.

```
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\007'
```

Si ROW FORMAT est omis, le format par défaut est DELIMITED FIELDS TERMINATED BY ’\A’ (début d’en-tête)et LINES TERMINATED BY ’\n’ (nouvelle ligne).

ROW FORMAT SERDE ’nom_sérialisation_désérialisation’ , [WITH SERDEPROPERTIES (’nom_propriété’ = ’valeur_propriété’ [, ...])]

Clause qui spécifie le format SERDE des données sous-jacentes.

’nom_sérialisation_désérialisation’

Le nom de l' SerDe. Vous pouvez spécifier les formats suivants :

- org.apache.hadoop.hive.serde2. RegexSerDe
- com.amazonaws.glue.serde. GrokSerDe
- org.apache.hadoop.hive.serde2.ouvrir CSVSerde

  Ce paramètre prend en charge la SerDe propriété suivante pour Open CSVSerde :

  ```
  'wholeFile' = 'true'
  ```

  Définissez la propriété `wholeFile` sur `true` pour analyser correctement les caractères de nouvelle ligne (\n) dans les chaînes entre guillemets pour les requêtes OpenCSV.
- org.openx.data.json. JsonSerDe
  - Le SERDE JSON prend également en charge les fichiers Ion.
  - Le JSON doit être dans un format correct.
  - Les horodatages dans Ion et JSON doivent utiliser le format ISO86 01.

- Ce paramètre prend en charge les SerDe propriétés suivantes pour JsonSerDe :

  ```
  'strip.outer.array'='true'
  ```

  Ion/JSON Traite les fichiers contenant un très grand tableau entre crochets ([...]) comme s'il contenait plusieurs enregistrements JSON dans le tableau.

- com.amazon.ionhiveserde. IonHiveSerDe

  Le format Amazon ION fournit des formats texte et binaire, en plus des types de données. Pour une table externe qui fait référence à des données au format ION, vous mappez chaque colonne de la table externe à l'élément correspondant dans les données au format ION. Pour plus d'informations, consultez Amazon Ion. Vous devez également spécifier les formats d'entrée et de sortie.

WITH SERDEPROPERTIES ('nom_propriété' = 'valeur_propriété' [, ...])]

À titre facultatif, spécifiez les noms et valeurs des propriétés, séparés par des virgules.

Si ROW FORMAT est omis, le format par défaut est DELIMITED FIELDS TERMINATED BY '\A' (début d'en-tête)et LINES TERMINATED BY '\n' (nouvelle ligne).

STORED AS format_fichier

Format des fichiers de données.

Les formats valides sont les suivants :

- PARQUET
- RCFILE (pour l'utilisation des données ColumnarSerDe uniquement, pas LazyBinaryColumnarSerDe)
- SEQUENCEFILE
- TEXTFILE (pour les fichiers texte, y compris les fichiers JSON).
- ORC
- AVRO
- INPUTFORMAT 'nom_classe_format_entrée' OUTPUTFORMAT 'nom_classe_format_sortie'

La commande CREATE EXTERNAL TABLE AS prend uniquement en charge deux formats de fichiers, TEXTFILE et PARQUET.

Pour INPUTFORMAT et OUTPUTFORMAT, indiquez un nom de classe comme dans l'exemple suivant.

```
'org.apache.hadoop.mapred.TextInputFormat'
```

LOCATION { 's3://bucket/folder/' | 's3://bucket/manifest_file'}

Chemin menant au compartiment Amazon S3 ou au dossier qui contient les fichiers de données ou un fichier manifeste qui contient une liste de chemins d'objets Amazon S3. Les compartiments doivent se trouver dans la même AWS région que le cluster Amazon Redshift. Pour obtenir une liste des régions AWS prises en charge, consultez Limites d'Amazon Redshift Spectrum.

Si le chemin précise un compartiment ou un dossier, par exemple, `'s3://amzn-s3-demo-bucket/custdata/'`, Redshift Spectrum analyse les fichiers qui se trouvent dans le compartiment ou dossier spécifié et dans tous les sous-dossiers. Redshift Spectrum ignore les fichiers masqués ainsi que les fichiers dont le nom commence par un point ou un trait de soulignement.

Si le chemin indique un fichier manifeste, l'argument `'s3://bucket/manifest_file'` doit explicitement faire référence à un seul fichier, par exemple, `'s3://amzn-s3-demo-bucket/manifest.txt'`. Il ne peut pas faire référence à un préfixe de clé.

Le manifeste est un fichier texte au format JSON qui répertorie l'URL de chaque fichier qui doit être chargé à partir d'Amazon S3, ainsi que la taille du fichier, en octets. L'URL inclut le nom du compartiment et le chemin d'objet complet du fichier. Les fichiers spécifiés dans le manifeste peuvent se trouver dans différents compartiments, mais tous les compartiments doivent se trouver dans la même AWS région que le cluster Amazon Redshift. Si un fichier est répertorié deux fois, le fichier est chargé deux fois. L'exemple suivant illustre le format JSON pour un manifeste qui charge trois fichiers.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket1/custdata.1", "meta": { "content_length":
 5956875 } },
    {"url":"s3://amzn-s3-demo-bucket1/custdata.2", "meta": { "content_length":
 5997091 } },
    {"url":"s3://amzn-s3-demo-bucket2/custdata.1", "meta": { "content_length":
 5978675 } }
  ]
}
```

Vous pouvez rendre obligatoire l'inclusion d'un fichier particulier. Pour ce faire, incluez une option `mandatory` au niveau du fichier dans le manifeste. Lorsque vous interrogez une table externe avec un fichier obligatoire manquant, l'instruction SELECT échoue. Assurez-vous que tous les fichiers inclus dans la définition de la table externe sont présents. S'ils ne sont pas tous présents, une erreur apparaît, indiquant le premier fichier obligatoire introuvable. L'exemple suivant montre le fichier JSON d'un manifeste dont l' `mandatory` option est définie sur `true`.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket1/custdata.1", "mandatory":true, "meta":
 { "content_length": 5956875 } },
    {"url":"s3://amzn-s3-demo-bucket1/custdata.2", "mandatory":false, "meta":
 { "content_length": 5997091 } },
    {"url":"s3://amzn-s3-demo-bucket2/custdata.1", "meta": { "content_length":
 5978675 } }
  ]
}
```

Pour référencer les fichiers créés à l'aide de la commande UNLOAD, vous devez utiliser le manifeste créé à l'aide de la commande UNLOAD avec le paramètre MANIFEST. Le fichier manifeste est compatible avec un fichier manifeste pour Commande COPY depuis Amazon S3, mais il n'utilise pas les mêmes clés. Les clés inutilisées sont ignorées.

TABLE PROPERTIES ('nom_propriété'='valeur_propriété' [, ...])

Clause qui définit la définition de table pour des propriétés de table.

> **Note**
>
> Les propriétés de table sont sensibles à la casse.

'compression_type'='valeur'

Propriété qui définit le type de compression à utiliser si le nom de fichier ne contient aucune extension. Si vous définissez cette propriété et que le nom de fichier contient une extension, celle-ci est ignorée et la valeur définie par la propriété est utilisée. Les valeurs valides pour chaque type de compression sont les suivantes :

- bzip2
- gzip

- none
- snappy

’data_cleansing_enabled’=’true / false’

Cette propriété définit si la gestion des données est activée pour la table. Lorsque ’data_cleansing_enabled’ est définie sur « true », la gestion des données est activée pour la table. Lorsque ’data_cleansing_enabled’ est définie sur « false », la gestion des données est désactivée pour la table. Vous trouverez ci-dessous la liste des propriétés de gestion des données au niveau de la table contrôlées par cette propriété :

- column_count_mismatch_handling
- invalid_char_handling
- numeric_overflow_handling
- replacement_char
- surplus_char_handling

Pour obtenir des exemples, consultez Exemples de gestion des données.

’invalid_char_handling’=’value’

Spécifie l’action à effectuer lorsque les résultats de la requête contiennent des valeurs de caractères UTF-8 non valides. Vous pouvez spécifier les actions suivantes :

DISABLED

N’effectue pas de gestion des caractères non valides.

FAIL

Annule les requêtes renvoyant des données contenant des valeurs UTF-8 non valides.

SET_TO_NULL

Remplace les valeurs UTF-8 non valides par null.

DROP_ROW

Remplace chaque valeur de la ligne par null.

REPLACE

Remplace le caractère non valide par le caractère de remplacement que vous spécifiez à l’aide de `replacement_char`.

’replacement_char’=’character’

Spécifie le caractère de remplacement à utiliser lorsque vous définissez `invalid_char_handling` sur `REPLACE`.

’numeric_overflow_handling’=’value’

Spécifie l’action à effectuer lorsque les données ORC contiennent un entier (par exemple, BIGINT ou int64) supérieur à la définition de colonne (par exemple, SMALLINT ou int16). Vous pouvez spécifier les actions suivantes :

DISABLED

La gestion des caractères non valide est désactivée.

FAIL

Annulez la requête lorsque les données contiennent des caractères non valides.

SET_TO_NULL

Définissez les caractères non valides sur null.

DROP_ROW

Définissez chaque valeur de la ligne sur null.

’surplus_bytes_handling’=’value’

Spécifie comment traiter les données chargées qui dépassent la longueur du type de données défini pour les colonnes contenant des données VARBYTE. Par défaut, Redshift Spectrum définit la valeur sur null pour les données qui dépassent la largeur de la colonne.

Vous pouvez spécifier les actions suivantes à effectuer lorsque la requête renvoie des données qui dépassent la longueur du type de données :

SET_TO_NULL

Remplace les données qui dépassent la largeur de colonne par null.

DISABLED

N’effectue pas de gestion des octets excédentaires.

FAIL

Annule les requêtes qui renvoient des données dépassant la largeur de colonne.

DROP_ROW

Supprime toutes les lignes qui contiennent des données qui dépassent la largeur de la colonne.

TRUNCATE

Supprime les caractères qui dépassent le nombre maximal de caractères défini pour la colonne.

’surplus_char_handling’=’value’

Spécifie comment gérer les données chargées qui dépassent la longueur du type de données défini pour les colonnes contenant des données VARCHAR, CHAR ou chaîne. Par défaut, Redshift Spectrum définit la valeur sur null pour les données qui dépassent la largeur de la colonne.

Vous pouvez spécifier les actions suivantes à effectuer lorsque la requête renvoie des données qui dépassent la largeur de la colonne :

SET_TO_NULL

Remplace les données qui dépassent la largeur de colonne par null.

DISABLED

N’effectue pas de gestion des caractères excédentaires.

FAIL

Annule les requêtes qui renvoient des données dépassant la largeur de colonne.

DROP_ROW

Remplace chaque valeur de la ligne par null.

TRUNCATE

Supprime les caractères qui dépassent le nombre maximal de caractères défini pour la colonne.

’column_count_mismatch_handling’=’value’

Indique si le fichier contient moins ou plus de valeurs pour une ligne que le nombre de colonnes spécifié dans la définition de la table externe. Cette propriété est disponible uniquement pour un format de fichier texte non compressé. Vous pouvez spécifier les actions suivantes :

DISABLED

La gestion des non-correspondances du nombre de colonnes est désactivée.

FAIL

Fait échouer la requête si la non-correspondance du nombre de colonnes est détectée.

SET_TO_NULL

Remplit les valeurs manquantes avec NULL et ignore les valeurs supplémentaires de chaque ligne.

DROP_ROW

Supprime de l'analyse toutes les lignes qui contiennent une erreur de non-correspondance du nombre de colonnes.

'numRows'='nombre_lignes'

Propriété qui définit la valeur numRows de la définition de table. Afin de mettre à jour explicitement les statistiques d'une table externe, définissez la propriété numRows pour indiquer la taille de la table. Amazon Redshift n'analyse pas les tables externes pour générer les statistiques de table utilisées par l'optimiseur de requête afin de générer un plan de requête. Si des statistiques de table ne sont pas définies pour une table externe, Amazon Redshift génère un plan d'exécution de requête d'après l'hypothèse selon laquelle les tables externes sont les tables les plus volumineuses et les tables locales les tables les plus petites.

'skip.header.line.count'='nombre_lignes'

Propriété qui définit le nombre de lignes à ignorer au début de chaque fichier source.

'serialization.null.format'=' '

Propriété qui spécifie que Spectrum doit renvoyer une valeur `NULL` lorsqu'il y a une correspondance exacte avec le texte fourni dans un champ.

'orc.schema.resolution'='type_mappage'

Propriété qui définit le type de mappage de colonne sur le mappage pour des tables qui utilisent le format de données ORC. Cette propriété est ignorée pour tous les autres formats de données.

Les valeurs valides pour le type de mappage de colonne sont les suivantes :

- name
- position

Si la propriété orc.schema.resolution est omise, les colonnes sont mappées par nom par défaut. Si la propriété orc.schema.resolution est définie sur une autre valeur que 'name' ou 'position', les colonnes sont mappées par position. Pour en savoir plus sur le mappage des colonnes, consultez Mappage de colonnes de table externe à des colonnes ORC.

> **Note**
>
> La commande COPY mappe aux fichiers de données ORC uniquement par position. La propriété de table orc.schema.resolution n'a aucun effet sur le comportement de la commande COPY.

'write.parallel'='on / off'

Propriété qui définit si la commande CREATE EXTERNAL TABLE AS doit écrire les données en parallèle. Par défaut, CREATE EXTERNAL TABLE AS écrit les données en parallèle dans plusieurs fichiers, en fonction du nombre de tranches du cluster. Par défaut, l'option est activée. Lorsque 'write.parallel' est désactivé, CREATE EXTERNAL TABLE AS écrit en série dans un ou plusieurs fichiers de données sur Amazon S3. Cette propriété de table s'applique également à toute instruction INSERT ultérieure dans la même table externe.

'write.maxfilesize.mb'='size'

Propriété qui définit la taille maximale (en Mo) de chaque fichier écrit dans Amazon S3 par la commande CREATE EXTERNAL TABLE AS. La taille doit être un entier valide compris entre 5 et 6 200. La taille maximale par défaut du fichier est de 6 200 Mo. Cette propriété de table s'applique également à toute instruction INSERT ultérieure dans la même table externe.

'write.kms.key.id'='value'

Vous pouvez spécifier une AWS Key Management Service clé pour activer le chiffrement côté serveur (SSE) pour les objets Amazon S3, dont la valeur est l'une des suivantes :

- `auto`pour utiliser la AWS KMS clé par défaut stockée dans le compartiment Amazon S3.
- kms-key que vous spécifiez pour chiffrer les données.

select_statement

Instruction qui insère une ou plusieurs lignes dans la table externe en définissant une requête. Toutes les lignes produites par la requête sont écrites dans Amazon S3 au format texte ou Parquet en fonction de la définition de la table.

Exemples

Divers exemples sont disponibles à la page Exemples.

Notes d'utilisation

Cette rubrique contient des notes d'utilisation pour CREATE EXTERNAL TABLE. Vous ne pouvez pas afficher les détails des tables Amazon Redshift Spectrum en utilisant les mêmes ressources que pour les tables Amazon Redshift standard, telles que PG_TABLE_DEF, STV_TBL_PERM, PG_CLASS ou information_schema. Si votre outil de Business Intelligence ou d'analyse ne reconnaît pas les tables externes Redshift Spectrum, configurez votre application de façon à interroger SVV_EXTERNAL_TABLES et SVV_EXTERNAL_COLUMNS.

CREATE EXTERNAL TABLE AS

Dans certains cas, vous pouvez exécuter la commande CREATE EXTERNAL TABLE AS sur un catalogue de AWS Glue données, un catalogue AWS Lake Formation externe ou un métastore Apache Hive. Dans ce cas, vous utilisez un rôle AWS Identity and Access Management (IAM) pour créer le schéma externe. Ce rôle IAM doit disposer d'autorisations de lecture et d'écriture sur Amazon S3.

Si vous utilisez un catalogue Lake Formation, le rôle IAM doit être autorisé à créer une table dans le catalogue. Dans ce cas, il doit également disposer de l'autorisation d'emplacement du lac de données sur le chemin Amazon S3 cible. Ce rôle IAM devient le propriétaire de la nouvelle table AWS Lake Formation .

Pour vous assurer que les noms de fichiers sont uniques, Amazon Redshift utilise le format suivant pour le nom de chaque fichier téléchargé dans Amazon S3 par défaut.

*<date>*_*<time>*_*<microseconds>*_*<query_id>*_*<slice-number>*_part_*<part-number>*.*<format>*.

Par exemple : `20200303_004509_810669_1007_0001_part_00.parquet`.

Tenez compte des éléments suivants lors de l'exécution de la commande CREATE EXTERNAL TABLE AS :

- L'emplacement Amazon S3 doit être vide.
- Amazon Redshift prend uniquement en charge les formats PARQUET et TEXTFILE lors de l'utilisation de la clause STORED AS.

- Vous n'avez pas besoin de définir une liste de définitions de colonne. Les noms de colonnes et les types de données de colonne de la nouvelle table externe sont dérivés directement de la requête SELECT.
- Vous n'avez pas besoin de définir le type de données de la colonne de partition dans la clause PARTITIONED BY. Si vous spécifiez une clé de partition, le nom de cette colonne doit exister dans le résultat de la requête SELECT. Lorsque vous avez plusieurs colonnes de partition, leur ordre dans la requête SELECT n'a pas d'importance. Amazon Redshift utilise l'ordre défini dans la clause PARTITIONED BY pour créer la table externe.
- Amazon Redshift partitionne automatiquement les fichiers de sortie dans des dossiers de partition en fonction des valeurs de clé de partition. Par défaut, Amazon Redshift supprime les colonnes de partition des fichiers de sortie.
- La clause 'delimiter' LINES TERMINATED BY n'est pas prise en charge.
- La clause 'serde_name' ROW FORMAT SERDE n'est pas prise en charge.
- L'utilisation de fichiers manifestes n'est pas prise en charge. Ainsi, vous ne pouvez pas définir la clause LOCATION sur un fichier manifeste dans Amazon S3.
- Amazon Redshift met automatiquement à jour la propriété de table 'numRows' à la fin de la commande.
- La propriété de table 'compression_type' accepte uniquement 'none' ou 'snappy' pour le format de fichier PARQUET.
- Amazon Redshift n'autorise pas la clause LIMIT dans la requête SELECT externe. Au lieu de cela, vous pouvez utiliser une clause LIMIT imbriquée.
- Vous pouvez utiliser STL_UNLOAD_LOG pour suivre les fichiers écrits dans Amazon S3 par chaque opération CREATE EXTERNAL TABLE AS.

Autorisations pour créer et interroger des table externes

Pour créer des tables externes, vérifiez que vous êtes le propriétaire du schéma externe ou un superutilisateur. Pour transférer la propriété d'un schéma externe, utilisez ALTER SCHEMA. L'exemple suivant remplace le propriétaire du schéma `spectrum_schema` par `newowner`.

```
alter schema spectrum_schema owner to newowner;
```

Pour exécuter une requête Redshift Spectrum, vous devez avoir les autorisations suivantes :

- Autorisations d'utilisation du schéma

- Autorisation de créer des tables temporaires dans la base de données actuelle

L'exemple suivant accorde l'autorisation d'utiliser le schéma `spectrum_schema` au groupe d'utilisateurs `spectrumusers`.

```
grant usage on schema spectrum_schema to group spectrumusers;
```

L'exemple suivant accorde une autorisation temporaire concernant la base de données `spectrumdb` au groupe d'utilisateurs `spectrumusers`.

```
grant temp on database spectrumdb to group spectrumusers;
```

Pseudocolonnes

Par défaut, Amazon Redshift crée les tables externes avec les pseudo-colonnes $path et $size. Sélectionnez ces colonnes pour afficher le chemin d'accès aux fichiers de données sur Amazon S3 et la taille des fichiers de données de chaque ligne retournée par une requête. Les noms de colonne $path et $size doivent être délimités par des guillemets doubles. Une clause SELECT * ne renvoie pas de pseudo-colonnes. Vous devez inclure explicitement les noms de colonne $path et $size dans votre requête, comme l'illustre l'exemple suivant.

```
select "$path", "$size"
from spectrum.sales_part
where saledate = '2008-12-01';
```

Vous pouvez désactiver la création de pseudo-colonnes d'une séance en définissant le paramètre de configuration spectrum_enable_pseudo_columns avec la valeur false.

> **Important**
>
> La sélection de $size ou $path entraîne des frais, car Redshift Spectrum analyse les fichiers de données dans Amazon S3 pour déterminer la taille du jeu de résultats. Pour plus d'informations, consultez la section Tarification Amazon Redshift.

Définition des options de gestion des données

Vous pouvez définir des paramètres de table pour spécifier la gestion des entrées pour les données interrogées dans les tables externes, notamment :

- Caractères excédentaires dans les colonnes contenant des données VARCHAR, CHAR et chaîne. Pour plus d'informations, consultez la propriété de table externe `surplus_char_handling`.
- Caractères non valides dans les colonnes contenant des données VARCHAR, CHAR et chaîne. Pour plus d'informations, consultez la propriété de table externe `invalid_char_handling`.
- Caractère de remplacement à utiliser lorsque vous spécifiez REPLACE pour la propriété de table externe `invalid_char_handling`.
- Gestion des débordements de distribution dans les colonnes contenant des données entières et décimales. Pour plus d'informations, consultez la propriété de table externe `numeric_overflow_handling`.
- Surplus_bytes_handling pour spécifier la gestion des entrées pour les octets excédentaires dans les colonnes contenant des données VARBYTE. Pour plus d'informations, consultez la propriété de table externe `surplus_bytes_handling`.

Exemples

L'exemple suivant crée une table nommée SALES dans le schéma externe Amazon Redshift nommé `spectrum`. Les données figurent dans des fichiers texte délimités par des tabulations. La clause TABLE PROPERTIES définit la valeur numRows sur 170 000 lignes.

Selon l'identité que vous utilisez pour exécuter CREATE EXTERNAL TABLE, vous devrez peut-être configurer des autorisations IAM. Il est recommandé d'associer des politiques d'autorisation à un rôle IAM, puis de l'attribuer à des utilisateurs et à des groupes, le cas échéant. Pour plus d'informations, consultez Identity and Access Management dans Amazon Redshift.

```
create external table spectrum.sales(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
saledate date,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
row format delimited
fields terminated by '\t'
stored as textfile
```

```
location 's3://redshift-downloads/tickit/spectrum/sales/'
table properties ('numRows'='170000');
```

L'exemple suivant crée une table qui utilise le JsonSerDe pour référencer les données au format JSON.

```
create external table spectrum.cloudtrail_json (
event_version int,
event_id bigint,
event_time timestamp,
event_type varchar(10),
awsregion varchar(20),
event_name varchar(max),
event_source varchar(max),
requesttime timestamp,
useragent varchar(max),
recipientaccountid bigint)
row format serde 'org.openx.data.jsonserde.JsonSerDe'
with serdeproperties (
'dots.in.keys' = 'true',
'mapping.requesttime' = 'requesttimestamp'
) location 's3://amzn-s3-demo-bucket/json/cloudtrail';
```

L’exemple CREATE EXTERNAL TABLE AS suivant crée une table externe non partitionnée. Ensuite, il écrit le résultat de la requête SELECT au format Apache Parquet à l’emplacement Amazon S3 cible.

```
CREATE EXTERNAL TABLE spectrum.lineitem
STORED AS parquet
LOCATION 'S3://amzn-s3-demo-bucket/cetas/lineitem/'
AS SELECT * FROM local_lineitem;
```

L’exemple suivant crée une table externe partitionnée et inclut les colonnes de partition dans la requête SELECT.

```
CREATE EXTERNAL TABLE spectrum.partitioned_lineitem
PARTITIONED BY (l_shipdate, l_shipmode)
STORED AS parquet
LOCATION 'S3://amzn-s3-demo-bucket/cetas/partitioned_lineitem/'
AS SELECT l_orderkey, l_shipmode, l_shipdate, l_partkey FROM local_table;
```

Pour obtenir la liste des bases de données existantes dans le catalogue de données externes, interrogez la vue système SVV_EXTERNAL_DATABASES.

```
select eskind,databasename,esoptions from svv_external_databases order by databasename;
```

```
eskind | databasename | esoptions
-------+--------------
+-------------------------------------------------------------------------------
     1 | default      | {"REGION":"us-
west-2","IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
     1 | sampledb     | {"REGION":"us-
west-2","IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
     1 | spectrumdb   | {"REGION":"us-
west-2","IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
```

Pour afficher les détails des tables externes, interrogez les vues système SVV_EXTERNAL_TABLES et SVV_EXTERNAL_COLUMNS.

L'exemple suivant interroge la vue SVV_EXTERNAL_TABLES.

```
select schemaname, tablename, location from svv_external_tables;
```

```
schemaname | tablename            | location
-----------+----------------------
+-----------------------------------------------------
spectrum   | sales                | s3://redshift-downloads/tickit/spectrum/sales
spectrum   | sales_part           | s3://redshift-downloads/tickit/spectrum/
sales_partition
```

L'exemple suivant interroge la vue SVV_EXTERNAL_COLUMNS.

```
select * from svv_external_columns where schemaname like 'spectrum%' and tablename
 ='sales';
```

```
schemaname | tablename | columnname | external_type | columnnum | part_key
-----------+-----------+------------+---------------+-----------+---------
spectrum   | sales     | salesid    | int           |         1 |        0
spectrum   | sales     | listid     | int           |         2 |        0
spectrum   | sales     | sellerid   | int           |         3 |        0
spectrum   | sales     | buyerid    | int           |         4 |        0
```

```
spectrum   | sales     | eventid    | int           |         5 |        0
spectrum   | sales     | saledate   | date          |         6 |        0
spectrum   | sales     | qtysold    | smallint      |         7 |        0
spectrum   | sales     | pricepaid  | decimal(8,2)  |         8 |        0
spectrum   | sales     | commission | decimal(8,2)  |         9 |        0
spectrum   | sales     | saletime   | timestamp     |        10 |        0
```

Pour afficher les partitions de table, utilisez la requête suivante.

```
select schemaname, tablename, values, location
from svv_external_partitions
where tablename = 'sales_part';
```

```
schemaname | tablename  | values         | location
-----------+------------+----------------
+-------------------------------------------------------------------------
spectrum   | sales_part | ["2008-01-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-01
spectrum   | sales_part | ["2008-02-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-02
spectrum   | sales_part | ["2008-03-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-03
spectrum   | sales_part | ["2008-04-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-04
spectrum   | sales_part | ["2008-05-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-05
spectrum   | sales_part | ["2008-06-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-06
spectrum   | sales_part | ["2008-07-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-07
spectrum   | sales_part | ["2008-08-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-08
spectrum   | sales_part | ["2008-09-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-09
spectrum   | sales_part | ["2008-10-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-10
spectrum   | sales_part | ["2008-11-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-11
spectrum   | sales_part | ["2008-12-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-12
```

L’exemple suivant renvoie la taille totale des fichiers de données associés pour une table externe.

```
select distinct "$path", "$size"
   from spectrum.sales_part;

 $path                                                                      | $size
----------------------------------------------------------------------------+-------
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/ |  1616
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/ |  1444
s3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/ |  1444
```

Exemple de partitionnement

Exécutez la commande suivante pour créer une table externe partitionnée par date.

```
create external table spectrum.sales_part(
salesid integer,
listid integer,
sellerid integer,
buyerid integer,
eventid integer,
dateid smallint,
qtysold smallint,
pricepaid decimal(8,2),
commission decimal(8,2),
saletime timestamp)
partitioned by (saledate date)
row format delimited
fields terminated by '|'
stored as textfile
location 's3://redshift-downloads/tickit/spectrum/sales_partition/'
table properties ('numRows'='170000');
```

Exécutez les commandes ALTER TABLE suivantes pour ajouter les partitions.

```
alter table spectrum.sales_part
add if not exists partition (saledate='2008-01-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-01/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-02-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-02/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-03-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-03/';
```

```
alter table spectrum.sales_part
add if not exists partition (saledate='2008-04-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-04/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-05-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-05/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-06-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-06/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-07-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-07/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-08-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-08/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-09-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-09/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-10-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-10/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-11-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-11/';
alter table spectrum.sales_part
add if not exists partition (saledate='2008-12-01')
location 's3://redshift-downloads/tickit/spectrum/sales_partition/saledate=2008-12/';
```

Pour sélectionner les données de la table partitionnée, exécutez la requête suivante.

```
select top 10 spectrum.sales_part.eventid, sum(spectrum.sales_part.pricepaid)
from spectrum.sales_part, event
where spectrum.sales_part.eventid = event.eventid
  and spectrum.sales_part.pricepaid > 30
  and saledate = '2008-12-01'
group by spectrum.sales_part.eventid
order by 2 desc;
```

```
eventid | sum
--------+---------
    914 | 36173.00
   5478 | 27303.00
   5061 | 26383.00
```

```
   4406 | 26252.00
   5324 | 24015.00
   1829 | 23911.00
   3601 | 23616.00
   3665 | 23214.00
   6069 | 22869.00
   5638 | 22551.00
```

Pour afficher les partitions de la table externe, interrogez la vue système SVV_EXTERNAL_PARTITIONS.

```
select schemaname, tablename, values, location from svv_external_partitions
where tablename = 'sales_part';
```

```
schemaname | tablename  | values         | location
-----------+------------+----------------
+-----------------------------------------------
spectrum   | sales_part | ["2008-01-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-01
spectrum   | sales_part | ["2008-02-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-02
spectrum   | sales_part | ["2008-03-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-03
spectrum   | sales_part | ["2008-04-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-04
spectrum   | sales_part | ["2008-05-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-05
spectrum   | sales_part | ["2008-06-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-06
spectrum   | sales_part | ["2008-07-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-07
spectrum   | sales_part | ["2008-08-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-08
spectrum   | sales_part | ["2008-09-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-09
spectrum   | sales_part | ["2008-10-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-10
spectrum   | sales_part | ["2008-11-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-11
spectrum   | sales_part | ["2008-12-01"] | s3://redshift-downloads/tickit/spectrum/
sales_partition/saledate=2008-12
```

Exemples de format de ligne

L'exemple suivant illustre la spécification des paramètres ROW FORMAT SERDE pour les fichiers de données stockés au format AVRO.

```
create external table spectrum.sales(salesid int, listid int, sellerid int,
 buyerid int, eventid int, dateid int, qtysold int, pricepaid decimal(8,2), comment
 VARCHAR(255))
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.avro.AvroSerDe'
WITH SERDEPROPERTIES ('avro.schema.literal'='{\"namespace\": \"dory.sample\",\"name\":
 \"dory_avro\",\"type\": \"record\", \"fields\": [{\"name\":\"salesid\", \"type\":\"int
\"},
{\"name\":\"listid\", \"type\":\"int\"},
{\"name\":\"sellerid\", \"type\":\"int\"},
{\"name\":\"buyerid\", \"type\":\"int\"},
{\"name\":\"eventid\",\"type\":\"int\"},
{\"name\":\"dateid\",\"type\":\"int\"},
{\"name\":\"qtysold\",\"type\":\"int\"},
{\"name\":\"pricepaid\", \"type\": {\"type\": \"bytes\", \"logicalType\": \"decimal\",
 \"precision\": 8, \"scale\": 2}}, {\"name\":\"comment\",\"type\":\"string\"}]}')
STORED AS AVRO
location 's3://amzn-s3-demo-bucket/avro/sales' ;
```

Voici un exemple de spécification des paramètres ROW FORMAT SERDE à l'aide RegEx de.

```
create external table spectrum.types(
cbigint bigint,
cbigint_null bigint,
cint int,
cint_null int)
row format serde 'org.apache.hadoop.hive.serde2.RegexSerDe'
with serdeproperties ('input.regex'='([^\\x01]+)\\x01([^\\x01]+)\\x01([^\\x01]+)\
\x01([^\\x01]+)')
stored as textfile
location 's3://amzn-s3-demo-bucket/regex/types';
```

L'exemple suivant illustre la spécification des paramètres ROW FORMAT SERDE à l'aide de Grok.

```
create external table spectrum.grok_log(
timestamp varchar(255),
pid varchar(255),
loglevel varchar(255),
```

```
progname varchar(255),
message varchar(255))
row format serde 'com.amazonaws.glue.serde.GrokSerDe'
with serdeproperties ('input.format'='[DFEWI], \\[%{TIMESTAMP_ISO8601:timestamp} #
%{POSINT:pid:int}\\] *(?<loglevel>:DEBUG|FATAL|ERROR|WARN|INFO) -- +%{DATA:progname}:
 %{GREEDYDATA:message}')
stored as textfile
location 's3://DOC-EXAMPLE-BUCKET/grok/logs';
```

L'exemple suivant montre un exemple de définition d'un journal d'accès au serveur Amazon S3 dans un compartiment S3. Vous pouvez utiliser Redshift Spectrum pour interroger les journaux d'accès Amazon S3.

```
CREATE EXTERNAL TABLE spectrum.mybucket_s3_logs(
bucketowner varchar(255),
bucket varchar(255),
requestdatetime varchar(2000),
remoteip varchar(255),
requester varchar(255),
requested varchar(255),
operation varchar(255),
key varchar(255),
requesturi_operation varchar(255),
requesturi_key varchar(255),
requesturi_httpprotoversion varchar(255),
httpstatus varchar(255),
errorcode varchar(255),
bytessent bigint,
objectsize bigint,
totaltime varchar(255),
turnaroundtime varchar(255),
referrer varchar(255),
useragent varchar(255),
versionid varchar(255)
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
'input.regex' = '([^ ]*) ([^ ]*) \\[(.*?)\\] ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 \"([^ ]*)\\s*([^ ]*)\\s*([^ ]*)\" (- |[^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 ([^ ]*) (\"[^\"]*\") ([^ ]*).*$')
LOCATION 's3://amzn-s3-demo-bucket/s3logs';
```

L'exemple suivant illustre la spécification des paramètres ROW FORMAT SERDE pour les données au format ION.

```
CREATE EXTERNAL TABLE tbl_name (columns)
ROW FORMAT SERDE 'com.amazon.ionhiveserde.IonHiveSerDe'
STORED AS
INPUTFORMAT 'com.amazon.ionhiveserde.formats.IonInputFormat'
OUTPUTFORMAT 'com.amazon.ionhiveserde.formats.IonOutputFormat'
LOCATION 's3://amzn-s3-demo-bucket/prefix'
```

Exemples de gestion des données

Les exemples suivants permettent d'accéder au fichier : spi_global_rankings.csv. Vous pouvez charger le fichier `spi_global_rankings.csv` dans un compartiment Amazon S3 pour essayer ces exemples.

L'exemple suivant permet de créer le schéma externe `schema_spectrum_uddh` et la base de données `spectrum_db_uddh`. Pour`aws-account-id`, entrez votre identifiant de AWS compte et pour `role-name` entrez le nom de votre rôle Redshift Spectrum.

```
create external schema schema_spectrum_uddh
from data catalog
database 'spectrum_db_uddh'
iam_role 'arn:aws:iam::aws-account-id:role/role-name'
create external database if not exists;
```

L'exemple suivant permet de créer une table externe `soccer_league` dans le schéma externe `schema_spectrum_uddh`.

```
CREATE EXTERNAL TABLE schema_spectrum_uddh.soccer_league
(
  league_rank smallint,
  prev_rank   smallint,
  club_name   varchar(15),
  league_name varchar(20),
  league_off  decimal(6,2),
  league_def  decimal(6,2),
  league_spi  decimal(6,2),
  league_nspi integer
)
```

```
ROW FORMAT DELIMITED
    FIELDS TERMINATED BY ','
    LINES TERMINATED BY '\n\l'
stored as textfile
LOCATION 's3://spectrum-uddh/league/'
table properties ('skip.header.line.count'='1');
```

Vérifiez le nombre de lignes dans la table `soccer_league`.

```
select count(*) from schema_spectrum_uddh.soccer_league;
```

Le nombre de lignes s'affiche.

```
count
645
```

La requête suivante affiche les 10 premiers clubs. Comme le club `Barcelona` présente un caractère non valide dans la chaîne, une valeur NULL s'affiche pour le nom.

```
select league_rank,club_name,league_name,league_nspi
from schema_spectrum_uddh.soccer_league
where league_rank between 1 and 10;
```

```
league_rank club_name league_name   league_nspi
1  Manchester City Barclays Premier Lea  34595
2  Bayern Munich German Bundesliga  34151
3  Liverpool Barclays Premier Lea  33223
4  Chelsea  Barclays Premier Lea  32808
5  Ajax  Dutch Eredivisie  32790
6  Atletico  Madrid Spanish Primera Divi 31517
7  Real Madrid Spanish Primera Divi  31469
8  NULL         Spanish Primera Divi            31321
9  RB Leipzig German Bundesliga  31014
10  Paris Saint-Ger French Ligue 1   30929
```

L'exemple suivant modifie la table `soccer_league` pour spécifier les propriétés de la table externe `invalid_char_handling`, `replacement_char` et `data_cleansing_enabled` afin d'insérer un point d'interrogation (?) pour remplacer les caractères inattendus.

```
alter  table schema_spectrum_uddh.soccer_league
```

```
set table properties
 ('invalid_char_handling'='REPLACE','replacement_char'='?','data_cleansing_enabled'='true');
```

L'exemple suivant montre comment interroger la table `soccer_league` pour les équipes dont le rang est compris entre 1 et 10.

```
select league_rank,club_name,league_name,league_nspi
from schema_spectrum_uddh.soccer_league
where league_rank between 1 and 10;
```

Comme les propriétés de la table ont été modifiées, les résultats affichent les 10 premiers clubs, avec le point d'interrogation (?) comme caractère de remplacement dans la huitième ligne pour le club `Barcelona`.

```
league_rank club_name league_name  league_nspi
1  Manchester City Barclays Premier Lea 34595
2  Bayern Munich German Bundesliga 34151
3  Liverpool Barclays Premier Lea 33223
4  Chelsea  Barclays Premier Lea 32808
5  Ajax  Dutch Eredivisie 32790
6  Atletico Madrid Spanish Primera Divi 31517
7  Real Madrid Spanish Primera Divi 31469
8  Barcel?na Spanish Primera Divi 31321
9  RB Leipzig German Bundesliga 31014
10  Paris Saint-Ger French Ligue 1  30929
```

L'exemple suivant modifie la table `soccer_league` pour spécifier les propriétés de la table externe `invalid_char_handling` afin de supprimer les lignes contenant des caractères inattendus.

```
alter table schema_spectrum_uddh.soccer_league
set table properties
 ('invalid_char_handling'='DROP_ROW','data_cleansing_enabled'='true');
```

L'exemple suivant montre comment interroger la table `soccer_league` pour les équipes dont le rang est compris entre 1 et 10.

```
select league_rank,club_name,league_name,league_nspi
from schema_spectrum_uddh.soccer_league
where league_rank between 1 and 10;
```

Les résultats affichent les premiers clubs, sans inclure la huitième ligne pour le club `Barcelona`.

```
league_rank   club_name         league_name            league_nspi
1             Manchester City   Barclays Premier Lea   34595
2             Bayern Munich     German Bundesliga      34151
3             Liverpool         Barclays Premier Lea   33223
4             Chelsea           Barclays Premier Lea   32808
5             Ajax              Dutch Eredivisie       32790
6             Atletico Madrid   Spanish Primera Divi   31517
7             Real Madrid       Spanish Primera Divi   31469
9             RB Leipzig        German Bundesliga      31014
10            Paris Saint-Ger   French Ligue 1         30929
```

CRÉER UNE VUE EXTERNE

La fonctionnalité des vues du catalogue de données en version préliminaire est disponible uniquement dans les régions suivantes.

- USA Est (Ohio) (us-east-2)
- USA Est (Virginie du Nord) (us-east-1)
- USA Ouest (Californie du Nord) (us-west-1)
- Asie-Pacifique (Tokyo) (ap-northeast-1)
- Europe (Irlande) (eu-west-1)
- Europe (Stockholm) (eu-north-1)

Crée une vue du catalogue de données. Les vues du catalogue de données constituent un schéma de vues unique qui fonctionne parfaitement avec d'autres moteurs SQL tels qu'Amazon Athena et Amazon EMR. Vous pouvez interroger la vue depuis le moteur de votre choix. Pour plus d'informations sur les vues du catalogue de données, voir Création de vues du catalogue de données.

Syntaxe

```
CREATE EXTERNAL VIEW schema_name.view_name [ IF NOT EXISTS ]
{catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name |
 external_schema_name.view_name}
AS query_definition;
```

Paramètres

schema_name.view_name

Le schéma attaché à votre AWS Glue base de données, suivi du nom de la vue.

PROTECTED

Spécifie que la commande CREATE EXTERNAL VIEW ne doit se terminer que si la requête contenue dans query_definition peut être terminée avec succès.

IF NOT EXISTS

Crée la vue si elle n'existe pas déjà.

catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name | external_schema_name.view_name

Notation du schéma à utiliser lors de la création de la vue. Vous pouvez spécifier d'utiliser une base de AWS Glue Data Catalog données Glue que vous avez créée ou un schéma externe que vous avez créé. Consultez CREATE DATABASE et CREATE EXTERNAL SCHEMA pour plus d'informations.

query_definition

Définition de la requête SQL exécutée par Amazon Redshift pour modifier la vue.

Exemples

L'exemple suivant crée une vue du catalogue de données nommée sample_schema.glue_data_catalog_view.

```
CREATE EXTERNAL PROTECTED VIEW sample_schema.glue_data_catalog_view IF NOT EXISTS
AS SELECT * FROM sample_database.remote_table "remote-table-name";
```

CREATE FUNCTION

Permet de créer une fonction scalaire définie par l'utilisateur à l'aide d'une clause SELECT SQL ou d'un programme Python.

Pour plus d'informations et d'exemples, consultez Fonctions définies par l'utilisateur dans Amazon Redshift.

Privilèges requis

Vous devez être autorisé par l'une des méthodes suivantes pour exécuter CREATE OR REPLACE FUNCTION :

- Pour CREATE FUNCTION :
  - Le super-utilisateur peut utiliser à la fois des langages fiables et non fiables pour créer des fonctions.
  - Les utilisateurs disposant du privilège CREATE [OR REPLACE] FUNCTION peuvent créer des fonctions avec des langages fiables.
- Pour REPLACE FUNCTION :
  - Superuser
  - Utilisateurs disposant du privilège CREATE [OR REPLACE] FUNCTION
  - Propriétaire de la fonction

Syntaxe

```
CREATE [ OR REPLACE ] FUNCTION f_function_name
( { [py_arg_name  py_arg_data_type |
sql_arg_data_type } [ , ... ] ] )
RETURNS data_type
{ VOLATILE | STABLE | IMMUTABLE }
AS $$
  { python_program | SELECT_clause }
$$ LANGUAGE { plpythonu | sql }
```

Paramètres

OR REPLACE

Spécifie que si une fonction ayant le même nom et les mêmes types de données pour les arguments en entrée, ou signature, existe déjà, la fonction existante est remplacée. Vous pouvez uniquement remplacer une fonction par une nouvelle fonction qui définit un ensemble identique de types de données. Vous devez être un super-utilisateur pour remplacer une fonction.

Si vous définissez une fonction avec le même nom qu'une fonction existante, mais avec une signature différente, vous créez une nouvelle fonction. En d'autres termes, le nom de la fonction est surchargé. Pour plus d'informations, consultez Surcharge des noms de fonctions.

f_function_name

Nom de la fonction. Si vous spécifiez un nom de schéma (tel que `myschema.myfunction`), la fonction est créée à l'aide du schéma spécifié. Sinon, la fonction est créée dans le schéma en cours. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

Nous vous recommandons d'utiliser le préfixe `f_` pour tous les noms de fonctions UDF. Comme Amazon Redshift réserve le préfixe `f_` pour les noms UDF, si vous utilisez le préfixe `f_`, vous vous assurez que votre nom UDF n'est pas conflit avec un nom existant ou futur de fonction SQL intégré à Amazon Redshift. Pour plus d'informations, consultez Prévention des conflits de dénomination des fonctions UDF.

Vous pouvez définir plus d'une fonction portant le même nom de fonction si les types de données des arguments en entrée sont différents. En d'autres termes, le nom de la fonction est surchargé. Pour de plus amples informations, veuillez consulter Surcharge des noms de fonctions.

py_arg_name py_arg_data_type | sql_arg_data_type

Pour une fonction Python définie par l'utilisateur, une liste des noms d'arguments en entrée et des types de données. Pour une fonction SQL définie par l'utilisateur, une liste des types de données, sans nom d'argument. Dans une fonction Python définie par l'utilisateur, faites référence aux arguments à l'aide des noms d'arguments. Dans une fonction SQL définie par l'utilisateur, faites référence aux arguments à l'aide de $1, $2, et ainsi de suite, en fonction de l'ordre des arguments dans la liste d'arguments.

Pour une fonction SQL définie par l'utilisateur, les types de données d'entrée et de retour peuvent être de n'importe quel type de données Amazon Redshift standard. Pour une fonction Python définie par l'utilisateur, les types de données entrant et de retour peuvent être SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE PRECISION, BOOLEAN, CHAR, VARCHAR, DATE ou TIMESTAMP. De plus, les fonctions définies par l'utilisateur en Python (UDFs) prennent en charge le type de données ANYELEMENT. Ce type de données est automatiquement converti en un type de données standard en fonction du type de données de l'argument correspondant fourni lors de l'exécution. Si plusieurs arguments utilisent ANYELEMENT, ils se résolvent dans le même type de données lors de l'exécution, en fonction du premier argument ANYELEMENT de la liste. Pour plus d'informations, consultez Types de données de fonctions Python définies par l'utilisateur et Types de données.

Vous pouvez spécifier un maximum de 32 arguments.

RETURNS type_données

Type de données de la valeur renvoyée par la fonction. Le type de données RETURNS peut être n'importe quel type de données Amazon Redshift standard. En outre, Python UDFs peut utiliser le type de données ANYELEMENT, qui est automatiquement converti en un type de données standard en fonction de l'argument fourni lors de l'exécution. Si vous spécifiez ANYELEMENT pour le type de données retournées, au moins un argument doit utiliser ANYELEMENT. Le type de données retournées correspond au type de données fourni pour l'argument ANYELEMENT lors de l'appel de la fonction. Pour plus d'informations, consultez Types de données de fonctions Python définies par l'utilisateur.

VOLATILE | STABLE | IMMUTABLE

Informe l'optimiseur de requête à propos de l'instabilité de la fonction.

Vous obtiendrez la meilleure optimisation possible si vous qualifiez votre fonction avec la catégorie d'instabilité la plus stricte qui s'y applique. Toutefois, si la catégorie est trop stricte, il existe un risque que l'optimiseur ignore par erreur certains appels, d'où un jeu de résultats incorrect. En matière de rigueur, en commençant par la moins stricte, les catégories d'instabilité sont les suivantes :

- VOLATILE
- STABLE
- IMMUTABLE

VOLATILE

Soit les mêmes arguments, la fonction peut renvoyer des résultats différents sur des appels successifs, y compris pour les lignes d'une même instruction. Comme l'optimiseur de requête ne peut pas effectuer d'hypothèse sur le comportement d'une fonction volatile, une requête qui utilise une fonction volatile doit réévaluer la fonction pour chaque ligne d'entrée.

STABLE

Soit les mêmes arguments, la fonction est garantie renvoyer les mêmes résultats pour toutes les lignes traitées au sein d'une même instruction. La fonction peut renvoyer des résultats différents lorsqu'elle est appelée dans différents instructions. Cette catégorie permet à l'optimiseur d'optimiser plusieurs appels de la fonction au sein d'une même instruction en un seul appel de l'instruction.

IMMUTABLE

Soit les mêmes arguments, la fonction renvoie toujours le même résultat, indéfiniment. Quand une requête appelle une fonction `IMMUTABLE` avec des arguments constants, l'optimiseur pré-évalue la fonction.

AS $$ instruction $$

Construction qui englobe l'instruction à exécuter. Les mots-clés littéraux `AS $$` et `$$` sont obligatoires.

Amazon Redshift requiert que vous placiez l'instruction dans votre fonction à l'aide d'un format appelé guillemets dollar. Tout ce qui se trouve dans l'encadrement est passé exactement comme tel. Vous n'avez pas besoin de définir de séquence d'échappement pour les caractères spéciaux, car les contenus de la chaîne sont écrits littéralement.

Avec les guillemets dollar, vous utilisez une paire de symboles dollar ($$) pour marquer le début et la fin de l'instruction à exécuter, comme illustré dans l'exemple suivant.

```
$$ my statement $$
```

Le cas échéant, entre les symboles dollar de chaque paire, vous pouvez spécifier une chaîne pour aider à identifier l'instruction. La chaîne que vous utilisez doit être la même au début et à la fin des paires d'encadrement. Cette chaîne est sensible à la casse et suit les mêmes contraintes qu'un identificateur sans guillemets, sauf que celui-ci ne peut pas contenir de symboles dollar. L'exemple suivant utilise la chaîne `test`.

```
$test$ my statement $test$
```

Pour plus d'informations sur les guillemets dollar, consultez relative aux constantes de chaîne avec guillemet dollar dans Lexical Structure dans la documentation PostgreSQL.

python_program

Programme exécutable Python valide qui renvoie une valeur. L'instruction que vous transmettez avec la fonction doit être conforme aux exigences de mise en retrait telles que spécifiées dans le Guide de style pour Python Code sur le site web Python. Pour plus d'informations, consultez Support du langage Python pour UDFs.

clause_SQL

Clause SELECT SQL.

La clause SELECT ne peut pas inclure les types de clause suivants :

- FROM
- INTO
- WHERE
- GROUP BY
- ORDER BY
- LIMIT

LANGUAGE { plpythonu | sql }

Pour Python, spécifiez `plpythonu`. Pour SQL, spécifiez `sql`. Vous devez avoir l'autorisation pour USAGE ON LANGUAGE pour SQL ou plpythonu. Pour plus d'informations, consultez Sécurité et autorisations UDF.

Notes d'utilisation

Fonctions imbriquées

Vous pouvez appeler une autre fonction SQL définie par l'utilisateur à partir d'une fonction SQL définie par l'utilisateur. La fonction imbriquée doit exister lorsque vous exécutez la commande CREATE FUNCTION. Amazon Redshift ne suit pas les dépendances pour. Ainsi UDFs, si vous supprimez la fonction imbriquée, Amazon Redshift ne renvoie aucune erreur. Toutefois, la fonction définie par l'utilisateur échoue si la fonction imbriquée n'existe pas. Par exemple, la fonction suivante appelle la fonction `f_sql_greater` dans la clause SELECT.

```
create function f_sql_commission (float, float )
  returns float
stable
as $$
  select f_sql_greater ($1, $2)
$$ language sql;
```

Privilèges et sécurité des fonctions UDF

Pour créer une fonction définie par l'utilisateur, vous devez avoir l'autorisation pour USAGE ON LANGUAGE pour SQL ou plpythonu (Python). Par défaut, USAGE ON LANGUAGE SQL est accordé à PUBLIC. Toutefois, vous devez accorder explicitement USAGE ON LANGUAGE PLPYTHONU à des utilisateurs ou des groupes spécifiques.

Pour révoquer le privilège USAGE pour SQL, révoquez d'abord USAGE de PUBLIC. Accordez ensuite l'utilisation de SQL uniquement aux utilisateurs ou groupes spécifiques autorisés à créer du SQL UDFs. L'exemple suivant révoque le privilège USAGE pour SQL de PUBLIC, puis accorde USAGE au groupe d'utilisateurs `udf_devs`.

```
revoke usage on language sql from PUBLIC;
grant usage on language sql to group udf_devs;
```

Pour exécuter une fonction définie par l'utilisateur, vous devez disposer de l'autorisation d'exécution pour chaque fonction. Par défaut, l'autorisation d'exécution pour new UDFs est accordée à PUBLIC. Pour limiter l'utilisation, révoquez l'autorisation d'exécution de PUBLIC pour la fonction. Ensuite, accordez le privilège à des individus ou à des groupes spécifiques.

L'exemple ci-dessous révoque l'autorisation d'exécution de la fonction `f_py_greater` de PUBLIC, puis accorde USAGE au groupe d'utilisateurs `udf_devs`.

```
revoke execute on function f_py_greater(a float, b float) from PUBLIC;
grant execute on function f_py_greater(a float, b float) to group udf_devs;
```

Les super-utilisateurs ont tous les privilèges par défaut.

Pour plus d'informations, consultez GRANT et REVOKE.

Exemples

Exemple de fonction scalaire Python définie par l'utilisateur

L'exemple suivant crée une fonction Python définie par l'utilisateur qui compare deux entiers et renvoie la valeur la plus grande.

```
create function f_py_greater (a float, b float)
  returns float
stable
as $$
  if a > b:
    return a
  return b
$$ language plpythonu;
```

L'exemple suivant interroge la table SALES et appelle la nouvelle fonction `f_py_greater` pour renvoyer COMMISSION ou 20 % du PRICEPAID, quelle que soit la valeur la plus grande.

```
select f_py_greater (commission, pricepaid*0.20) from sales;
```

Exemple de fonction scalaire SQL définie par l'utilisateur

L'exemple suivant crée une fonction qui compare deux nombres et renvoie la valeur la plus grande.

```
create function f_sql_greater (float, float)
  returns float
stable
as $$
  select case when $1 > $2 then $1
    else $2
  end
$$ language sql;
```

La requête suivante appelle la nouvelle fonction `f_sql_greater` pour interroger la table SALES et renvoie COMMISSION ou 20 % du PRICEPAID, quelle que soit la valeur la plus grande.

```
select f_sql_greater (commission, pricepaid*0.20) from sales;
```

CREATE GROUP

Définit un nouveau groupe d'utilisateurs. Seul un super-utilisateur peut créer un groupe.

Syntaxe

```
CREATE GROUP group_name
[ [ WITH ] [ USER username ] [, ...] ]
```

Paramètres

nom_groupe

Nom du nouveau groupe d'utilisateurs. Les noms de groupe commençant par deux tirets bas sont réservés pour un usage Amazon Redshift interne. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

WITH

Syntaxe facultative pour indiquer des paramètres supplémentaires pour CREATE GROUP.

USER

Ajoutez un ou plusieurs utilisateurs au groupe.

nom d'utilisateur

Nom de l'utilisateur à ajouter au groupe.

Exemples

L'exemple suivant crée un groupe d'utilisateurs nommé ADMIN_GROUP avec deux utilisateurs, et. ADMIN1 ADMIN2

```
create group admin_group with user admin1, admin2;
```

CREATE IDENTITY PROVIDER

Définit un nouveau fournisseur d'identité. Seul un super-utilisateur peut créer un fournisseur d'identité.

Syntaxe

```
CREATE IDENTITY PROVIDER identity_provider_name TYPE type_name
NAMESPACE namespace_name
[PARAMETERS parameter_string]
[APPLICATION_ARN arn]
[IAM_ROLE iam_role]
[AUTO_CREATE_ROLES
    [ TRUE [ { INCLUDE | EXCLUDE } GROUPS LIKE filter_pattern] |
      FALSE
    ]
  ];
```

Paramètres

identity_provider_name

Nom du nouveau fournisseur d'identité. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

type_name

Fournisseur d’identité avec lequel interagir. Azure et AWSIDC sont actuellement les seuls fournisseurs d'identité pris en charge.

namespace_name

Espace de noms. Il s’agit d’un identifiant unique et abrégé pour le répertoire du fournisseur d’identité.

parameter_string

Chaîne contenant un objet JSON correctement formaté qui contient des paramètres et des valeurs requis pour le fournisseur d’identité.

arn

Le nom de ressource Amazon (ARN) d'une application gérée par IAM Identity Center. Ce paramètre n'est applicable que lorsque le type de fournisseur d'identité est. AWSIDC

iam_role

Rôle IAM qui fournit les autorisations nécessaires pour établir la connexion à IAM Identity Center. Ce paramètre n'est applicable que lorsque le type de fournisseur d'identité est. AWSIDC

auto_create_roles

Active ou désactive la fonctionnalité de création automatique des rôles. Si la valeur est VRAI, Amazon Redshift active la fonctionnalité de création automatique des rôles. Si la valeur est FALSE, Amazon Redshift désactive la fonctionnalité de création automatique des rôles. Si la valeur de ce paramètre n'est pas spécifiée, Amazon Redshift détermine la valeur selon la logique suivante :

- S'il `AUTO_CREATE_ROLES` est fourni mais que la valeur n'est pas spécifiée, la valeur est définie sur TRUE.
- Si `AUTO_CREATE_ROLES` ce n'est pas le cas et que le fournisseur d'identité l'est AWSIDC, la valeur est définie sur FALSE.
- Si `AUTO_CREATE_ROLES` ce n'est pas le cas et que le fournisseur d'identité est Azure, la valeur est définie sur TRUE.

Pour inclure des groupes, spécifiez`INCLUDE`. La valeur par défaut est vide, ce qui signifie inclure tous les groupes si cette option `AUTO_CREATE_ROLES` est activée.

Pour exclure des groupes, spécifiez`EXCLUDE`. La valeur par défaut est vide, ce qui signifie qu'il ne faut exclure aucun groupe si cette option `AUTO_CREATE_ROLES` est activée.

filter_pattern

Expression de caractères UTF-8 valide avec un modèle correspondant aux noms de groupes. L'option LIKE effectue une mise en correspondance sensible à la casse qui prend en charge les métacaractères de mise en correspondance de modèle suivants :

Métacaractère	Description
`%`	Met en correspondance une séquence de zéro ou plusieurs caractères.
`_`	Met en correspondance un seul caractère.

Si filter_pattern ne contient pas de métacaractères, le modèle représente uniquement la chaîne elle-même ; dans ce cas, LIKE a la même fonction que l'opérateur d'égalité.

filter_pattern prend en charge les caractères suivants :

- Caractères alphabétiques majuscules et minuscules (A-Z et a-z)
- Chiffres (0-9)
- Les caractères spéciaux suivants :

```
_ % ^ * + ? { } , $
```

Exemples

L'exemple suivant crée un fournisseur d'identité nommé oauth_standard, avec un TYPE azure, afin d'établir une communication avec Microsoft Azure Active Directory (AD).

```
CREATE IDENTITY PROVIDER oauth_standard TYPE azure
NAMESPACE 'aad'
PARAMETERS '{"issuer":"https://sts.windows.net/2sdfdsf-d475-420d-b5ac-667adad7c702/",
"client_id":"87f4aa26-78b7-410e-bf29-57b39929ef9a",
"client_secret":"BUAH~ewrqewrqwerUUY^%tHe1oNZShoiU7",
"audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift"]
}'
```

Vous pouvez connecter une application gérée par IAM Identity Center à un cluster provisionné existant ou à un groupe de travail Amazon Redshift Serverless. Cela vous permet de gérer l'accès à une base de données Redshift via IAM Identity Center. Pour ce faire, exécutez une commande SQL comme dans l'exemple suivant. Vous devez être administrateur de base de données.

```
CREATE IDENTITY PROVIDER "redshift-idc-app" TYPE AWSIDC
NAMESPACE 'awsidc'
APPLICATION_ARN 'arn:aws:sso::123456789012:application/ssoins-12345f67fe123d4/apl-
a0b0a12dc123b1a4'
IAM_ROLE 'arn:aws:iam::123456789012:role/MyRedshiftRole';
```

Dans ce cas, l'ARN de l'application identifie l'application gérée à laquelle se connecter. Vous pouvez le trouver en courant`SELECT * FROM SVV_IDENTITY_PROVIDERS;`.

Pour plus d'informations sur l'utilisation de CREATE IDENTITY PROVIDER, y compris des exemples supplémentaires, consultez Fédération de fournisseurs d'identité natifs pour Amazon Redshift. Pour plus d'informations sur la configuration d'une connexion à IAM Identity Center depuis Redshift, voir Connect Redshift à IAM Identity Center pour offrir aux utilisateurs une expérience d'authentification unique.

CREATE LIBRARY

Installe une bibliothèque Python, qui est disponible pour que les utilisateurs l'intègrent lors de la création d'une fonction définie par l'utilisateur (UDF) avec la commande CREATE FUNCTION. La taille totale des bibliothèques installées par l'utilisateur ne peut pas dépasser 100 Mo.

CREATE LIBRARY ne peut pas être exécutée à l'intérieur d'un bloc de transaction (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

Amazon Redshift prend en charge Python version 2.7. Pour plus d'informations, consultez la page www.python.org.

Pour de plus amples informations, veuillez consulter Exemple : importation de modules de bibliothèque Python personnalisés.

Privilèges requis

Les privilèges suivants sont requis pour CREATE LIBRARY :

- Superuser

- Utilisateurs disposant du privilège CREATE LIBRARY ou du privilège du langage spécifié

Syntaxe

```
CREATE [ OR REPLACE ] LIBRARY library_name LANGUAGE plpythonu
FROM
{ 'https://file_url'
| 's3://bucketname/file_name'
authorization
  [ REGION [AS] 'aws_region']
  IAM_ROLE { default | ‘arn:aws:iam::<Compte AWS-id>:role/<role-name>’ }
}
```

Paramètres

OR REPLACE

Spécifie que si une bibliothèque portant le même nom que celui-ci existe déjà, la bibliothèque existante est remplacée. REPLACE valide immédiatement. Si une fonction UDF qui dépend de la bibliothèque s’exécute simultanément, la fonction UDF peut échouer ou renvoyer des résultats inattendus, même si la fonction UDF est en cours d’exécution au sein d’une transaction. Vous devez être le propriétaire ou un super-utilisateur pour remplacer une bibliothèque.

nom_bibliothèque

Nom de la bibliothèque à installer. Vous ne pouvez pas créer une bibliothèque qui contient un module avec le même nom qu’un module de la bibliothèque standard Python ou qu’un module Amazon Redshift préinstallé. Si une bibliothèque existante installée par l’utilisateur emploie le même package Python que la bibliothèque à installer, vous devez supprimer la bibliothèque existante avant d’installer la nouvelle bibliothèque. Pour plus d'informations, consultez Support du langage Python pour UDFs.

LANGUAGE plpythonu

Langage à utiliser. Python (plpythonu) est le seul langage pris en charge. Amazon Redshift prend en charge Python version 2.7. Pour plus d’informations, consultez la page www.python.org.

FROM

Emplacement du fichier bibliothèque. Vous pouvez spécifier un compartiment Amazon S3 et le nom d’objet ou vous pouvez spécifier une URL pour télécharger le fichier depuis un site

web public. La bibliothèque doit être empaquetée sous la forme d'un fichier `.zip`. Pour plus d'informations, consultez Création et installation des modules Python dans la documentation Python.

https://url_fichier

URL pour télécharger le fichier depuis un site web public. L'URL peut contenir jusqu'à trois redirections. Voici un exemple d'URL de fichier.

```
'https://www.example.com/pylib.zip'
```

s3://nom_compartiment/nom_fichier

Chemin d'accès vers un objet Amazon S3 qui contient le fichier bibliothèque. Voici un exemple de chemin d'objet Amazon S3.

```
's3://amzn-s3-demo-bucket/my-pylib.zip'
```

Si vous spécifiez un compartiment Amazon S3, vous devez également fournir les informations d'identification pour un utilisateur AWS qui est autorisé à télécharger le fichier.

> **Important**
>
> Si le compartiment Amazon S3 ne réside pas dans la même AWS région que votre cluster Amazon Redshift, vous devez utiliser l'option REGION pour spécifier la AWS région dans laquelle se trouvent les données. La valeur de aws_region doit correspondre à une AWS région répertoriée dans le tableau de la description des REGION paramètres de la commande COPY.

authorization

Clause indiquant la méthode que votre cluster utilise pour l'authentification et l'autorisation afin d'accéder au compartiment Amazon S3 qui contient le fichier bibliothèque. Votre cluster doit avoir l'autorisation d'accéder au compartiment Amazon S3 avec les actions LIST et GET.

La syntaxe de l'autorisation est identique à celle de la commande COPY. Pour plus d'informations, consultez Paramètres d'autorisation.

```
IAM_ROLE { default | 'arn:aws:iam::<Compte AWS-id>:role/<role-name>'
```

Utilisez le mot clé par défaut pour qu'Amazon Redshift utilise le rôle IAM défini comme rôle par défaut et associé au cluster lorsque la commande CREATE LIBRARY s'exécute.

Utilisez l'Amazon Resource Name (ARN) d'un rôle IAM que votre cluster utilise pour l'authentification et l'autorisation. Si vous spécifiez IAM_ROLE, vous ne pouvez pas utiliser ACCESS_KEY_ID et SECRET_ACCESS_KEY, SESSION_TOKEN ni CREDENTIALS.

Le cas échéant, si le compartiment Amazon S3 utilise le chiffrement côté serveur, fournissez la clé de chiffrement dans la chaîne credentials-args. Si vous utilisez des informations d'identification de sécurité temporaires, fournissez le jeton temporaire dans la chaîne credentials-args.

Pour plus d'informations, consultez informations d'identification de sécurité temporaires.

REGION [AS] région_aws

AWS Région dans laquelle se trouve le compartiment Amazon S3. REGION est obligatoire lorsque le compartiment Amazon S3 ne se trouve pas dans la même AWS région que le cluster Amazon Redshift. La valeur de aws_region doit correspondre à une AWS région répertoriée dans le tableau de la description des REGION paramètres de la commande COPY.

Par défaut, CREATE LIBRARY suppose que le compartiment Amazon S3 est situé dans la même AWS région que le cluster Amazon Redshift.

Exemples

Les deux exemples suivants installent le module Python urlparse, qui est empaqueté dans un fichier nommé `urlparse3-1.0.3.zip`.

La commande suivante installe une bibliothèque UDF nommée `f_urlparse` à partir d'un package qui a été téléchargé sur un compartiment Amazon S3 situé dans la région USA Est.

```
create library f_urlparse
language plpythonu
from 's3://amzn-s3-demo-bucket/urlparse3-1.0.3.zip'
credentials 'aws_iam_role=arn:aws:iam::<aws-account-id>:role/<role-name>'
region as 'us-east-1';
```

L'exemple suivant installe une bibliothèque nommée `f_urlparse` à partir d'un fichier bibliothèque sur un site web.

```
create library f_urlparse
language plpythonu
from 'https://example.com/packages/urlparse3-1.0.3.zip';
```

CREATE MASKING POLICY

Crée une nouvelle politique de masquage dynamique des données pour masquer les données d'un format donné. Pour plus d'informations sur le masquage dynamique des données, consultez Masquage dynamique des données.

Les super-utilisateurs et les utilisateurs ou les rôles disposant du rôle sys:secadmin peuvent créer une politique de masquage.

Syntaxe

```
CREATE MASKING POLICY
   policy_name [IF NOT EXISTS]
   WITH (input_columns)
   USING (masking_expression);
```

Paramètres

policy_name

Nom de la politique de masquage. La politique de masquage ne peut pas avoir le même nom qu'une autre politique de masquage qui existe déjà dans la base de données.

input_columns

Un tuple de noms de colonnes au format (type col1, type col2 ...).

Les noms de colonne sont utilisés comme entrée pour l'expression du masquage. Les noms des colonnes ne doivent pas nécessairement correspondre aux noms des colonnes masquées, mais les types de données d'entrée et de sortie doivent correspondre.

masking_expression

Expression SQL utilisée pour transformer les colonnes cibles. Elle peut être écrite à l'aide de fonctions de manipulation des données, telles que des fonctions de manipulation de chaînes, ou en conjonction avec des fonctions définies par l'utilisateur écrites en SQL, Python ou avec AWS Lambda. Vous pouvez inclure un tuple d'expressions de colonne pour masquer les politiques

ayant plusieurs sorties. Si vous utilisez une constante comme expression de masquage, vous devez la convertir explicitement en un type correspondant au type d'entrée.

Vous devez disposer de l'autorisation USAGE sur toutes les fonctions définies par l'utilisateur que vous utilisez dans l'expression du masquage.

CREATE MATERIALIZED VIEW

Crée une vue matérialisée basée sur une ou plusieurs tables Amazon Redshift. Vous pouvez également baser les vues matérialisées sur des tables externes créées à l'aide de Spectrum ou d'une requête fédérée. Pour obtenir des informations sur Spectrum, consultez Amazon Redshift Spectrum. Pour obtenir des informations sur la requête fédérée, consultez Interrogation de données avec requête fédérée dans Amazon Redshift.

Syntaxe

```
CREATE MATERIALIZED VIEW mv_name
[ BACKUP { YES | NO } ]
[ table_attributes ]
[ AUTO REFRESH { YES | NO } ]
AS query
```

Paramètres

BACKUP

Clause qui indique si la vue matérialisée doit être incluse dans les instantanés de cluster automatisés et manuels.

Pour les vues matérialisées qui ne contiennent pas de données critiques, spécifiez BACKUP NO pour réduire le temps de traitement lors de la création d'instantanés et de la restauration à partir d'instantanés et pour réduire l'espace de stockage sur Amazon Simple Storage Service. Le paramètre BACKUP NO n'a aucun effet sur la réplication automatique des données vers les autres nœuds du cluster. Les vues matérialisées pour lesquelles BACKUP NO est spécifié sont donc restaurées en cas de défaillance d'un nœud. La valeur par défaut est BACKUP YES.

table_attributes

Clause qui spécifie la manière dont les données de la vue matérialisée sont distribuées, notamment :

- Style de distribution de la vue matérialisée, au format `DISTSTYLE { EVEN | ALL | KEY }`. Si vous omettez cette clause, le style de distribution est `EVEN`. Pour plus d'informations, consultez Styles de distribution.
- Clé de distribution de la vue matérialisée, au format `DISTKEY ( distkey_identifier )`. Pour plus d'informations, consultez Détermination des styles de distribution.
- Clé de tri de la vue matérialisée, au format `SORTKEY ( column_name [, ...] )`. Pour plus d'informations, consultez Clés de tri.

AS requête

Une instruction `SELECT` valide qui définit la vue matérialisée et son contenu. L'ensemble de résultats de la requête définit les colonnes et les lignes de la vue matérialisée. Pour obtenir des informations sur les limitations lors de la création de vues matérialisées, consultez Limites.

En outre, des constructions de langage SQL spécifiques utilisées dans la requête déterminent si la vue matérialisée peut être reproduite de manière incrémentielle ou complète. Pour obtenir des informations sur la méthode d'actualisation, consultez REFRESH MATERIALIZED VIEW. Pour obtenir des informations sur les limitations de l'actualisation progressive, consultez Limites d'actualisation incrémentielle.

Si la requête contient une commande SQL qui ne prend pas en charge l'actualisation incrémentielle, Amazon Redshift affiche un message indiquant que la vue matérialisée utilisera une actualisation complète. Le message peut ou non s'afficher, selon l'application cliente SQL. Cochez la colonne `state` de STV_MV_INFO pour voir le type d'actualisation utilisé par une vue matérialisée.

AUTO REFRESH

Clause qui définit si la vue matérialisée doit être automatiquement actualisée avec les dernières modifications de ses tables de base ou non. La valeur par défaut est `NO`. Pour plus d'informations, consultez Actualisation d'une vue matérialisée.

Notes d'utilisation

Pour créer une vue matérialisée, vous devez disposer des privilèges suivants :

- Privilèges CREATE pour un schéma.
- Privilège SELECT au niveau de la table ou d'une colonne sur les tables de base pour créer une vue matérialisée. Si vous disposez de privilèges au niveau de colonnes spécifiques, vous pouvez créer une vue matérialisée uniquement avec ces colonnes.

Actualisation incrémentielle des vues matérialisées dans un partage de données

Amazon Redshift prend en charge l'actualisation automatique et incrémentielle des vues matérialisées dans un partage de données client lorsque les tables de base sont partagées. L'actualisation incrémentielle est une opération au cours de laquelle Amazon Redshift identifie les modifications apportées à la table de base ou aux tables après l'actualisation précédente et met à jour uniquement les enregistrements correspondants dans la vue matérialisée. Cela s'exécute plus rapidement qu'une actualisation complète et améliore les performances de la charge de travail. Il n'est pas nécessaire de modifier la définition de votre vue matérialisée pour tirer parti de l'actualisation incrémentielle.

Il existe quelques limites à prendre en compte pour tirer parti de l'actualisation incrémentielle avec une vue matérialisée :

- La vue matérialisée ne doit référencer qu'une seule base de données, locale ou distante.
- L'actualisation incrémentielle n'est disponible que sur les nouvelles vues matérialisées. Par conséquent, vous devez supprimer les vues matérialisées existantes et les recréer pour qu'une actualisation incrémentielle ait lieu.

Pour plus d'informations sur la création de vues matérialisées dans un partage de données, consultez la section Utilisation des vues dans le cadre du partage de données Amazon Redshift, qui contient plusieurs exemples de requêtes.

Mises à jour DDL des vues matérialisées ou des tables de base

Lorsque vous utilisez des vues matérialisées dans Amazon Redshift, suivez ces notes d'utilisation pour les mises à jour du langage de définition de données (DDL) vers des vues matérialisées ou des tables de base.

- Vous pouvez ajouter des colonnes à une table de base sans affecter les vues matérialisées qui référencent cette table.
- Certaines opérations peuvent laisser la vue matérialisée dans un état qui ne peut pas du tout être actualisé. Par exemple, des opérations telles que le renommage ou la suppression d'une colonne, le modification d'un type de colonne et le changement de nom d'un schéma. Ce type de vue matérialisée peut être interrogé mais pas actualisé. Dans ce cas, vous devez annuler et recréer la vue matérialisée.
- En général, vous ne pouvez pas modifier la définition d'une vue matérialisée (son instruction SQL).

- Vous ne pouvez pas renommer une vue matérialisée.

Limites

Vous ne pouvez pas définir de vue matérialisée qui référence ou comprend l'un des éléments suivants :

- Vues standard ou tables et vues système.
- Tables temporaires.
- Fonctions définies par l'utilisateur.
- La clause ORDER BY, LIMIT ou OFFFET.
- Références contraignantes tardives aux tables de base. En d'autres termes, toute table de base ou colonne associée référencée dans la définition de la requête SQL de la vue matérialisée doit exister et être valide.
- Fonctions de nœud leader uniquement : CURRENT_SCHEMA, CURRENT_SCHEMAS, HAS_DATABASE_PRIVILEGE, HAS_SCHEMA_PRIVILEGE, HAS_TABLE_PRIVILEGE.
- Vous ne pouvez pas utiliser l'option AUTO REFRESH YES lorsque la définition de la vue matérialisée inclut des fonctions mutables ou des schémas externes. Vous ne pouvez pas non plus l'utiliser lorsque vous définissez une vue matérialisée sur une autre vue matérialisée.
- Vous n'avez pas besoin d'exécuter manuellement ANALYSE sur les vues matérialisées. Pour l'heure, l'analyse s'effectue uniquement via AUTO ANALYZE. Pour de plus amples informations, veuillez consulter Analyse des tables.
- Tables protégées par RLS ou DDM.

Exemples

L'exemple suivant montre comment créer une vue matérialisée à partir de trois tables de base qui sont jointes et agrégées. Chaque ligne représente une catégorie avec le nombre de billets vendus. Lorsque vous interrogez la vue matérialisée tickets_mv, vous accédez directement aux données précalculées dans la vue matérialisée tickets_mv.

```
CREATE MATERIALIZED VIEW tickets_mv AS
    select   catgroup,
    sum(qtysold) as sold
    from     category c, event e, sales s
    where    c.catid = e.catid
```

```
    and      e.eventid = s.eventid
    group by catgroup;
```

L'exemple suivant crée une vue matérialisée similaire à l'exemple précédent et utilise la fonction d'agrégation MAX().

```
CREATE MATERIALIZED VIEW tickets_mv_max AS
    select   catgroup,
    max(qtysold) as sold
    from     category c, event e, sales s
    where    c.catid = e.catid
    and      e.eventid = s.eventid
    group by catgroup;

SELECT name, state FROM STV_MV_INFO;
```

L'exemple suivant utilise une clause UNION ALL pour joindre la table `public_sales` Amazon Redshift et la table `spectrum.sales` Redshift Spectrum pour créer une vue matérialisée `mv_sales_vw`. Pour obtenir des informations sur la commande CREATE EXTERNAL TABLE pour Amazon Redshift Spectrum, consultez CREATE EXTERNAL TABLE. La table externe Redshift Spectrum fait référence aux données sur Amazon S3.

```
CREATE MATERIALIZED VIEW mv_sales_vw as
select salesid, qtysold, pricepaid, commission, saletime from public.sales
union all
select salesid, qtysold, pricepaid, commission, saletime from spectrum.sales
```

L'exemple suivant crée une vue matérialisée `mv_fq` basée sur une table externe de requête fédérée. Pour obtenir des informations sur la requête fédérée, consultez CREATE EXTERNAL SCHEMA.

```
CREATE MATERIALIZED VIEW mv_fq as select firstname, lastname from apg.mv_fq_example;

select firstname, lastname from mv_fq;
 firstname | lastname
-----------+----------
 John      | Day
 Jane      | Doe
(2 rows)
```

L'exemple suivant montre la définition d'une vue matérialisée.

```
SELECT pg_catalog.pg_get_viewdef('mv_sales_vw'::regclass::oid, true);

pg_get_viewdef
-------------------------------------------------
create materialized view mv_sales_vw as select a from t;
```

L'exemple suivant montre comment définir AUTO REFRESH dans la définition de la vue matérialisée et indique également un DISTSTYLE. Commencez par créer une table de base simple.

```
CREATE TABLE baseball_table (ball int, bat int);
```

Créez ensuite une vue matérialisée.

```
CREATE MATERIALIZED VIEW mv_baseball DISTSTYLE ALL AUTO REFRESH YES AS SELECT ball AS
 baseball FROM baseball_table;
```

Vous pouvez maintenant interroger la vue matérialisée mv_baseball. Pour vérifier si AUTO REFRESH est activé pour une vue matérialisée, consultez STV_MV_INFO.

L'exemple suivant crée une vue matérialisée qui fait référence à une table source dans une autre base de données. Cela suppose que la base de données contenant la table source, database_A, se trouve dans le même cluster ou groupe de travail que votre vue matérialisée, que vous créez dans database_B. (Vous pouvez remplacer vos propres bases de données par l'exemple.) Tout d'abord, créez une table dans database_A appelée cities pour les villes, avec une colonne cityname pour les noms de villes. Définissez le type de données de la colonne comme VARCHAR. Après avoir créé la table source, exécutez la commande suivante dans database_B pour créer une vue matérialisée dont la source est votre table cities. Assurez-vous de spécifier la base de données et le schéma de la table source dans la clause FROM :

```
CREATE MATERIALIZED VIEW cities_mv AS
SELECT  cityname
FROM    database_A.public.cities;
```

Interrogez la vue matérialisée que vous avez créée. La requête extrait les enregistrements dont la source d'origine est la table cities dans database_A :

```
select * from cities_mv;
```

Lorsque vous exécutez l'instruction SELECT, cities_mv renvoie les enregistrements. Les enregistrements sont actualisés à partir de la table source uniquement lorsqu'une instruction REFRESH est exécutée. Notez également que vous ne pouvez pas mettre à jour les enregistrements directement dans la vue matérialisée. Pour plus d'informations sur l'actualisation des données dans une vue matérialisée, consultez REFRESH MATERIALIZED VIEW.

Pour en savoir plus sur la présentation des vues matérialisées et les commandes SQL utilisées pour actualiser et supprimer les vues matérialisées, consultez les rubriques suivantes :

- Vues matérialisées dans Amazon Redshift
- REFRESH MATERIALIZED VIEW
- DROP MATERIALIZED VIEW

CREATE MODEL

Rubriques

- Prérequis
- Privilèges requis
- Contrôle des coûts
- CREATE MODEL
- Paramètres
- Notes d'utilisation
- Cas d'utilisation

Prérequis

Avant d'utiliser l'instruction CREATE MODEL, suivez les étapes requises détaillées dans Configuration du cluster pour l'utilisation d'Amazon Redshift ML. Vous trouverez ci-dessous un résumé général des étapes requises.

- Créez un cluster Amazon Redshift à l'aide de la console de AWS gestion ou de l'interface de ligne de AWS commande (AWS CLI).
- Attachez la politique AWS Identity and Access Management (IAM) lors de la création du cluster.
- Pour permettre à Amazon Redshift et à SageMaker AI d'assumer le rôle d'interaction avec d'autres services, ajoutez la politique de confiance appropriée au rôle IAM.

Pour plus d'informations sur le rôle IAM, la politique d'approbation et d'autres prérequis, consultez Configuration du cluster pour l'utilisation d'Amazon Redshift ML.

Vous trouverez ci-dessous différents cas d'utilisation pour l'instruction CREATE MODEL.

- CREATE MODEL simple
- CREATE MODEL avec guide de l'utilisateur
- CRÉEZ des XGBoost modèles avec AUTO OFF
- Modèle BYOM (Bring Your Own Model) : inférence locale
- Modèle BYOM (Bring Your Own Model) : inférence distante
- CREATE MODEL avec K-MEANS
- CREATE MODEL

Privilèges requis

Les privilèges suivants sont requis pour CREATE MODEL :

- Superuser
- Utilisateurs disposant du privilège CREATE MODEL
- Rôles avec le privilège GRANT CREATE MODEL

Contrôle des coûts

Amazon Redshift ML utilise les ressources existantes du cluster pour créer des modèles de prédiction, de sorte que vous n'avez pas à payer de frais supplémentaires. Cependant, des coûts supplémentaires peuvent s'appliquer si vous devez redimensionner votre cluster ou si vous souhaitez entraîner vos modèles. Amazon Redshift ML utilise Amazon SageMaker AI pour entraîner des modèles, ce qui entraîne un coût supplémentaire. Il existe des moyens de contrôler les coûts supplémentaires, par exemple en limitant la durée maximale de l'entraînement ou en limitant le nombre d'exemples d'entraînement utilisés pour entraîner votre modèle. Pour obtenir plus d'informations, consultez Costs for using Amazon Redshift ML (Coûts d'utilisation d'Amazon Redshift ML).

CREATE MODEL

Voici un résumé des options de base de la syntaxe CREATE MODEL complète.

Syntaxe CREATE MODEL complète

Voici la syntaxe complète de l'instruction CREATE MODEL.

> **Important**
>
> Lorsque vous créez un modèle à l'aide de l'instruction CREATE MODEL, suivez l'ordre des mots-clés dans la syntaxe suivante.

```
CREATE MODEL model_name
FROM { table_name | ( select_statement )  | 'job_name' }
[ TARGET column_name ]
FUNCTION function_name [ ( data_type [, ...] ) ]
[ RETURNS data_type ]
  -- supported only for BYOM
[ SAGEMAKER 'endpoint_name'[:'model_name']]
  -- supported only for BYOM remote inference
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
[ AUTO ON / OFF ]
  -- default is AUTO ON
[ MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER | KMEANS | FORECAST } ]
  -- not required for non AUTO OFF case, default is the list of all supported types
  -- required for AUTO OFF
[ PROBLEM_TYPE ( REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION ) ]
  -- not supported when AUTO OFF
[ OBJECTIVE ( 'MSE' | 'Accuracy' | 'F1' | 'F1_Macro' | 'AUC' |
             'reg:squarederror' | 'reg:squaredlogerror'| 'reg:logistic'|
             'reg:pseudohubererror' | 'reg:tweedie' | 'binary:logistic' |
 'binary:hinge',
             'multi:softmax' | 'RMSE' | 'WAPE' | 'MAPE' | 'MASE' |
 'AverageWeightedQuantileLoss' ) ]
  -- for AUTO ON: first 5 are valid
  -- for AUTO OFF: 6-13 are valid
  -- for FORECAST: 14-18 are valid
[ PREPROCESSORS 'string' ]
  -- required for AUTO OFF, when it has to be 'none'
  -- optional for AUTO ON
[ HYPERPARAMETERS { DEFAULT | DEFAULT EXCEPT ( Key 'value' (,...) ) } ]
  -- support XGBoost hyperparameters, except OBJECTIVE
  -- required and only allowed for AUTO OFF
  -- default NUM_ROUND is 100
  -- NUM_CLASS is required if objective is multi:softmax (only possible for AUTO OFF)
```

```
[ SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket',  |
    -- required
 TAGS 'string', |
    -- optional
 KMS_KEY_ID 'kms_string', |
    -- optional
 S3_GARBAGE_COLLECT on / off, |
    -- optional, defualt is on.
 MAX_CELLS integer, |
    -- optional, default is 1,000,000
 MAX_RUNTIME integer (, ...) |
    -- optional, default is 5400 (1.5 hours)
 HORIZON integer, |
    -- required if creating a forecast model
 FREQUENCY integer, |
    -- required if creating a forecast model
 PERCENTILES string, |
    -- optional if creating a forecast model
 MAX_BATCH_ROWS integer -- optional for BYOM remote inference
    ) ]
```

Paramètres

model_name

Nom du modèle. Le nom du modèle d'un schéma doit être unique.

FROM { table_name | (select_query) | 'job_name'}

Le table_name ou la requête qui spécifie les données d'entraînement. Il peut s'agir soit d'une table existante dans le système, soit d'une requête SELECT compatible avec Amazon RedShift entre parenthèses, c'est-à-dire (). Le résultat de la requête doit contenir au moins deux colonnes.

TARGET column_name

Nom de la colonne qui devient la cible de la prédiction. La colonne doit exister dans la clause FROM.

FUNCTION function_name (data_type [, ...])

Nom de la fonction à créer et types de données des arguments d'entrée. Vous pouvez fournir le nom d'un schéma dans votre base de données au lieu d'un nom de fonction.

RETURNS type_données

Type de données à renvoyer par la fonction du modèle. Le type de SUPER données renvoyé s'applique uniquement au BYOM avec inférence à distance.

SAGEMAKER ’endpoint_name’[:’model_name’]

Nom du point de terminaison Amazon SageMaker AI. Si le nom du point de terminaison pointe vers un point de terminaison multimodèle, ajoutez le nom du modèle à utiliser. Le point de terminaison doit être hébergé au même Région AWS endroit que le cluster Amazon Redshift.

IAM_ROLE { default | ’arn:aws:iam::<account-id>:role/<role-name>’}

Utilisez le mot clé par défaut pour qu’Amazon Redshift utilise le rôle IAM défini comme rôle par défaut et associé au cluster lorsque la commande CREATE MODEL s’exécute. Vous pouvez également spécifier l’ARN d’un rôle IAM pour utiliser ce rôle.

[AUTO ON / OFF]

Active ou désactive la détection automatique CREATE MODEL du préprocesseur, de l’algorithme et de la sélection d’hyperparamètres. Lorsque vous créez un modèle de prévision, vous devez utiliser un AutoPredictor, où Amazon Forecast applique les combinaisons optimales d'algorithmes à chaque série chronologique de votre ensemble de données.

MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER | KMEANS | FORECAST }

(Facultatif) Spécifie le type de modèle. Vous pouvez spécifier si vous souhaitez entraîner un modèle d'un type de modèle spécifique, tel que le perceptron multicouche (MLP) XGBoost, le KMEANS ou le Linear Learner, qui sont tous des algorithmes pris en charge par Amazon AI Autopilot. SageMaker Si vous ne spécifiez pas le paramètre, tous les types de modèles pris en charge sont recherchés pendant l’entraînement pour trouver le meilleur modèle. Vous pouvez également créer un modèle de prévision dans Redshift ML pour créer des prévisions chronologiques précises.

PROBLEM_TYPE (REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION)

(Facultatif) Spécifie le type de problème. Si vous connaissez le type de problème, vous pouvez limiter la recherche Amazon Redshift au meilleur modèle de ce type en particulier. Si vous ne spécifiez pas ce paramètre, un type de problème est détecté pendant l’entraînement, en fonction de vos données.

OBJECTIF (« MSE » | « Précision » | « F1 » | « F1Macro » | « AUC » | « reg:squarederror » | « reg:squaredlogerror » | « reg:logistic » | « reg:pseudohubererror » | « reg:tweedie » | « binary:hinge » | « multi:softmax » | « RM » (« | » WAPE » | « MAPE » | « MASE » | « ») AverageWeightedQuantileLoss

(Facultatif) Spécifie le nom de la métrique d'objectif utilisée pour mesurer la qualité prédictive d'un système de machine learning. Cette métrique est optimisée pendant l'entraînement afin de fournir la meilleure estimation des valeurs des paramètres du modèle à partir des données. Si vous ne spécifiez pas de métrique explicitement, le comportement par défaut consiste à utiliser automatiquement MSE : pour la régression, F1 : pour la classification binaire, Précision : pour la classification multiclasse. Pour plus d'informations sur les objectifs, consultez Auto MLJob Objective dans le manuel Amazon SageMaker AI API Reference et Learning task parameters dans la documentation XGBOOST. Les valeurs RMSE, WAPE, MAPE, MASE et AverageWeightedQuantileLoss ne s'appliquent qu'aux modèles Forecast. Pour plus d'informations, consultez l'opération API CreateAutoPredictor.

PREPROCESSORS 'string'

(Facultatif) Spécifie certaines combinaisons de préprocesseurs à certains ensembles de colonnes. Le format est une liste de columnSets et les transformations appropriées à appliquer à chaque ensemble de colonnes. Amazon Redshift applique tous les transformateurs d'une liste de transformateurs spécifique à toutes les colonnes de la liste correspondante. ColumnSet Par exemple, pour appliquer OneHotEncoder avec Imputer aux colonnes t1 et t2, utilisez l'exemple de commande suivant.

```
CREATE MODEL customer_churn
FROM customer_data
TARGET 'Churn'
FUNCTION predict_churn
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
PROBLEM_TYPE BINARY_CLASSIFICATION
OBJECTIVE 'F1'
PREPROCESSORS '[
...
  {"ColumnSet": [
      "t1",
      "t2"
    ],
    "Transformers": [
      "OneHotEncoder",
      "Imputer"
```

```
    ]
  },
  {"ColumnSet": [
      "t3"
    ],
    "Transformers": [
      "OneHotEncoder"
    ]
  },
  {"ColumnSet": [
      "temp"
    ],
    "Transformers": [
      "Imputer",
      "NumericPassthrough"
    ]
  }
]'
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket'
)
```

HYPERPARAMETERS { DEFAULT | DEFAULT EXCEPT (key ‘value’ (,..)) }

Spécifie si les XGBoost paramètres par défaut sont utilisés ou remplacés par des valeurs spécifiées par l'utilisateur. Les valeurs doivent être entre guillemets simples. Vous trouverez ci-dessous des exemples de paramètres pour XGBoost et leurs valeurs par défaut.

Nom du paramètre	Valeur de paramètre	Valeur par défaut	Remarques
num_class	Entier	Requis pour la classification multiclasse.	N/A
num_round	Entier	100	N/A

Nom du paramètre	Valeur de paramètre	Valeur par défaut	Remarques
tree_method	Chaîne	Auto	N/A
max_depth	Entier	6	[0 , 10]
min_child_weight	Float	1	MinValue: 0, MaxValue 120
subsample	Float	1	MinValue: 0,5, MaxValue : 1
gamma	Float	0	MinValue: 0, MaxValue 5
alpha	Float	0	MinValue: 0, MaxValue 100
eta	Float	0.3	MinValue: 0,1, MaxValue 0,5
colsample_byleve	Float	1	MinValue: 0,1, MaxValue : 1
colsample_bynode	Float	1	MinValue: 0,1, MaxValue : 1
colsample_bytree	Float	1	MinValue: 0,5, MaxValue : 1
lambda	Float	1	MinValue: 0, MaxValue 100
max_delta_step	Entier	0	[0, 10]

PARAMÈTRES (S3_BUCKET 'amzn-s3-demo-bucket', | TAGS 'string', | KMS_KEY_ID 'kms_string', | S3_GARBAGE_COLLECT activé/désactivé, | MAX_CELLS entier, | MAX_RUNTIME (,...), | HORIZON entier, | FREQUENCY forecast_frequency, | tableau de chaînes PERCENTILES)

La clause S3_BUCKET spécifie l’emplacement Amazon S3 utilisé pour stocker les résultats intermédiaires.

(Facultatif) Le paramètre TAGS est une liste séparée par des virgules de paires clé-valeur que vous pouvez utiliser pour étiqueter les ressources créées dans Amazon SageMaker AI et Amazon Forecast. Les balises permettent d'organiser les ressources et d'imputer les coûts. Les valeurs de la paire étant facultatives, vous pouvez créer des balises en utilisant le format `key=value` ou en créant simplement une clé. Pour en savoir plus sur les balises dans Amazon Redshift, consultez Présentation du balisage.

(Facultatif) KMS_KEY_ID indique si Amazon Redshift utilise le chiffrement côté serveur avec une clé pour protéger les données au repos. AWS KMS Les données en transit sont protégées par le protocole SSL.

(Facultatif) La clause S3_GARBAGE_COLLECT { ON | OFF } spécifie si Amazon Redshift lance le récupérateur de mémoire sur les jeux de données obtenus utilisés pour entraîner les modèles et sur les modèles. Si la valeur est définie sur OFF, les jeux de données obtenus utilisés pour entraîner les modèles et les modèles eux-mêmes restent dans Amazon S3 et peuvent être utilisés à d'autres fins. Si la valeur est définie sur ON, Amazon Redshift supprime les artefacts dans Amazon S3 une fois l'entraînement terminé. La valeur par défaut est ON.

(Facultatif) La clause MAX_CELLS spécifie le nombre de cellules dans les données d'entraînement. Cette valeur est le produit du nombre d'enregistrements (dans la requête d'entraînement ou dans la table) multiplié par le nombre de colonnes. La valeur par défaut est 1 000 000.

(Facultatif) La clause MAX_RUNTIME spécifie la durée maximale d'entraînement. Les tâches d'entraînement se terminent souvent plus tôt en fonction de la taille du jeu de données. Ce nombre spécifie la durée maximale de l'entraînement. La valeur par défaut est 5 400 (90 minutes).

HORIZON spécifie le nombre maximum de prédictions que le modèle de prévision peut renvoyer. Une fois le modèle entraîné, vous ne pouvez pas modifier cet entier. Ce paramètre est obligatoire pour l'entraînement d'un modèle de prévision.

FREQUENCY indique le degré de précision en unités de temps que vous souhaitez pour les prévisions. Les options disponibles sont `Y | M | W | D | H | 30min | 15min | 10min | 5min | 1min`. Ce paramètre est obligatoire pour l'entraînement d'un modèle de prévision.

(Facultatif) PERCENTILES est une chaîne délimitée par des virgules qui spécifie les types de prévisions utilisés pour entraîner un prédicteur. Les types de prévisions peuvent être des quantiles compris entre 0,01 et 0,99, par incréments de 0,01 ou plus. Vous pouvez également spécifier la prévision moyenne à l'aide de la moyenne. Vous pouvez spécifier un maximum de cinq types de prévisions.

MAX_BATCH_ROWS entier

(Facultatif) Le nombre maximum de lignes qu'Amazon Redshift envoie dans une seule demande par lot pour un seul appel d' SageMaker IA. Il n'est pris en charge que pour le BYOM avec inférence à distance. La valeur minimale pour ce paramètre est 1. La valeur maximale est `INT_MAX` de 2 147 483 647. Ce paramètre n'est obligatoire que lorsque les types de données en entrée et en sortie sont de type SUPER. La valeur par défaut est`INT_MAX`, ou 2 147 483 647.

Notes d'utilisation

Tenez compte des éléments suivants lorsque vous utilisez CREATE MODEL :

- L'instruction CREATE MODEL fonctionne en mode asynchrone et renvoie le résultat de l'exportation des données d'entraînement vers Amazon S3. Les étapes restantes de la formation dans Amazon SageMaker AI se déroulent en arrière-plan. Bien que l'entraînement soit en cours, la fonction d'inférence correspondante est visible mais ne peut pas être exécutée. Vous pouvez interroger STV_ML_MODEL_INFO pour voir l'état de l'entraînement.
- L'entraînement peut s'exécuter jusqu'à 90 minutes en arrière-plan, par défaut dans le modèle Auto et peut être prolongée. Pour annuler l'entraînement, exécutez simplement la commande DROP MODEL.
- Le cluster Amazon Redshift que vous utilisez pour créer le modèle et le compartiment Amazon S3 utilisé pour mettre en page les données d'entraînement et les artefacts de modèle doivent se trouver dans la même région AWS .
- Pendant la formation du modèle, Amazon Redshift et SageMaker AI stockent les artefacts intermédiaires dans le compartiment Amazon S3 que vous fournissez. Par défaut, Amazon Redshift lance le récupérateur de mémoire à la fin de l'opération CREATE MODEL. Amazon Redshift supprime ces objets d'Amazon S3. Pour retenir ces artefacts sur Amazon S3, définissez l'option S3_GARBAGE COLLECT OFF.
- Vous devez utiliser au moins 500 lignes dans les données d'entraînement fournies dans la clause FROM.
- Vous pouvez spécifier jusqu'à 256 colonnes de fonctions (entrée) dans la clause FROM { table_name | (select_query) } lorsque vous utilisez l'instruction CREATE MODEL.
- Pour AUTO ON, les types de colonnes que vous pouvez utiliser comme jeu d'entraînement sont SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE, BOOLEAN, CHAR, VARCHAR, DATE, TIME, TIMETZ, TIMESTAMP et TIMESTAMPTZ. Pour AUTO OFF, les types de colonnes

que vous pouvez utiliser comme jeu d'entraînement sont SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, DOUBLE et BOOLEAN.

- Vous ne pouvez pas utiliser DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP, TIMESTAMPTZ, GEOMETRY, GEOGRAPHY, HLLSKETCH, SUPER ou VARBYTE comme type de colonne cible.
- Pour améliorer la précision du modèle, optez pour l'une des solutions suivantes :
  - Ajoutez autant de colonnes pertinentes que possible dans la commande CREATE MODEL lorsque vous spécifiez les données d'entraînement dans la clause FROM.
  - Utilisez une valeur plus grande pour MAX_RUNTIME et MAX_CELLS. Des valeurs plus élevées pour ce paramètre augmentent le coût de l'entraînement d'un modèle.
- L'exécution de l'instruction CREATE MODEL renvoie dès que les données d'entraînement sont calculées et exportées vers le compartiment Amazon S3. Ensuite, vous pouvez vérifier l'état de l'entraînement à l'aide de la commande SHOW MODEL. Lorsqu'un modèle entraîné en arrière-plan échoue, vous pouvez vérifier l'erreur à l'aide de la commande SHOW MODEL. Vous ne pouvez pas effectuer de nouvelle tentative pou un modèle ayant échoué. Utilisez la commande DROP MODEL pour supprimer un modèle ayant échoué et créer un nouveau modèle. Pour plus d'informations sur les modèles SHOW MODEL, consultez SHOW MODEL.
- Le modèle BYOM local prend en charge le même type de modèles que ceux pris en charge par le ML d'Amazon Redshift pour les cas non-BYOM. Amazon Redshift prend en charge les modèles simples XGBoost (XGBoost version 1.0 ou ultérieure), les modèles KMEANS sans préprocesseurs et les modèles XGBOOST/MLP/Linear Learner entraînés par Trained by Amazon AI Autopilot. SageMaker Il prend en charge ce dernier avec des préprocesseurs spécifiés par Autopilot et également pris en charge par Amazon SageMaker AI Neo.
- Si le routage amélioré de votre cluster Amazon Redshift est activé pour votre cloud privé virtuel (VPC), assurez-vous de créer un point de terminaison Amazon S3 VPC et un point de terminaison VPC AI pour le SageMaker VPC dans lequel se trouve votre cluster. Cela permet au trafic de s'exécuter via votre VPC entre ces services pendant l'exécution de la commande CREATE MODEL. Pour plus d'informations, consultez SageMaker AI Clarify Job Amazon VPC Subnets and Security Groups.

Cas d'utilisation

Les cas d'utilisation suivants montrent comment utiliser la clause CREATE MODEL en fonction de vos besoins.

CREATE MODEL simple

Voici un résumé des options de base de la syntaxe CREATE MODEL.

Syntaxe CREATE MODEL simple

```
CREATE MODEL model_name
FROM { table_name | ( select_query ) }
TARGET column_name
FUNCTION prediction_function_name
IAM_ROLE { default }
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket',
  [ MAX_CELLS integer ]
)
```

Paramètres CREATE MODEL simples

model_name

Nom du modèle. Le nom du modèle d'un schéma doit être unique.

FROM { table_name | (select_query) }

Le table_name ou la requête qui spécifie les données d'entraînement. Il peut s'agir soit d'une table existante dans le système, soit d'une requête SELECT compatible avec Amazon RedShift entre parenthèses, c'est-à-dire (). Le résultat de la requête doit contenir au moins deux colonnes.

TARGET column_name

Nom de la colonne qui devient la cible de la prédiction. La colonne doit exister dans la clause FROM.

FUNCTION prediction_function_name

Valeur qui spécifie le nom de la fonction de machine learning Amazon Redshift à générer par le modèle CREATE MODEL et utilisée pour effectuer des prédictions à l'aide de ce modèle. La fonction est créée dans le même schéma que l'objet de modèle et peut être surchargée.

L'apprentissage automatique Amazon Redshift prend en charge des modèles tels que les modèles Xtreme Gradient Boosted tree (XGBoost) pour la régression et la classification.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }

Utilisez le mot clé par défaut pour qu'Amazon Redshift utilise le rôle IAM défini comme rôle par défaut et associé au cluster lorsque la commande CREATE MODEL s'exécute. Vous pouvez également spécifier l'ARN d'un rôle IAM pour utiliser ce rôle.

S3_BUCKET 'amzn-s3-demo-bucket'

Le nom du compartiment Amazon S3 que vous avez créé précédemment était utilisé pour partager des données d'entraînement et des artefacts entre Amazon Redshift et SageMaker AI. Amazon Redshift crée un sous-dossier dans ce compartiment avant le déchargement des données d'entraînement. Lorsque l'entraînement est terminé, Amazon Redshift supprime le sous-dossier créé et son contenu.

MAX_CELLS integer

Nombre maximal de cellules à exporter à partir de la clause FROM. La valeur par défaut est 1 000 000.

Le nombre de cellules est le produit du nombre de lignes dans les données d'entraînement (produites par la table ou la requête de clause FROM) multiplié par le nombre de colonnes. Si le nombre de cellules dans les données d'entraînement est supérieur à celui spécifié par le paramètre max_cells, CREATE MODEL crée des sous-échantillons des données d'entraînement de la clause FROM pour réduire la taille du jeu de données d'entraînement en dessous de la valeur MAX_CELLS. Autoriser des jeux de données d'entraînement plus volumineux peut permettre de bénéficier d'une plus grande précision, mais peut également augmenter la durée et le coût d'entraînement du modèle.

Pour obtenir des informations sur les coûts d'utilisation d'Amazon Redshift, consultez Coûts d'utilisation d'Amazon Redshift ML.

Pour plus d'informations sur les coûts associés aux différents nombres de cellules et sur les détails de l'essai gratuit, consultez Tarification Amazon Redshift.

CREATE MODEL avec guide de l'utilisateur

Vous trouverez ci-dessous une description des options de CREATE MODEL en plus des options décrites dans CREATE MODEL simple.

Par défaut, CREATE MODEL recherche la meilleure combinaison de prétraitement et le meilleur modèle pour votre jeu de données spécifique. Il se peut que vous ayez besoin d'un contrôle plus

poussé ou d'introduire des connaissances de domaine supplémentaires (comme le type de problème ou l'objectif) pour votre modèle. Dans un scénario de désabonnement client, si le résultat « client n'est pas actif » est rare, l'objectif F1 est souvent préféré à l'objectif Précision. Étant donné que les modèles à haute précision peuvent prédire « le client est actif » tout le temps, il en résulte une haute précision, mais peu de valeur opérationnelle. Pour plus d'informations sur l'objectif F1, consultez Auto MLJob Objective dans le manuel Amazon SageMaker AI API Reference.

Ensuite, le modèle CREATE suit vos suggestions sur les aspects spécifiés, tels que l'objectif. Dans le même temps, CREATE MODEL détecte automatiquement les meilleurs préprocesseurs et les meilleurs hyperparamètres.

CREATE MODEL avec syntaxe de guide de l'utilisateur

CREATE MODEL offre plus de flexibilité sur les aspects que vous pouvez spécifier et les aspects qu'Amazon Redshift détecte automatiquement.

```
CREATE MODEL model_name
FROM { table_name | ( select_statement ) }
TARGET column_name
FUNCTION function_name
IAM_ROLE { default }
[ MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER} ]
[ PROBLEM_TYPE ( REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION ) ]
[ OBJECTIVE ( 'MSE' | 'Accuracy' | 'F1' | 'F1Macro' | 'AUC') ]
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket', |
  S3_GARBAGE_COLLECT { ON | OFF }, |
  KMS_KEY_ID 'kms_key_id', |
  MAX_CELLS integer, |
  MAX_RUNTIME integer (, ...)
)
```

CREATE MODEL avec paramètres de guide de l'utilisateur

MODEL_TYPE { XGBOOST | MLP | LINEAR_LEARNER }

(Facultatif) Spécifie le type de modèle. Vous pouvez spécifier si vous souhaitez entraîner un modèle d'un type de modèle spécifique XGBoost, tel que le perceptron multicouche (MLP) ou le Linear Learner, qui sont tous des algorithmes pris en charge par Amazon SageMaker AI Autopilot. Si vous ne spécifiez pas le paramètre, tous les types de modèles pris en charge sont recherchés pendant l'entraînement pour trouver le meilleur modèle.

PROBLEM_TYPE (REGRESSION | BINARY_CLASSIFICATION | MULTICLASS_CLASSIFICATION)

(Facultatif) Spécifie le type de problème. Si vous connaissez le type de problème, vous pouvez limiter la recherche Amazon Redshift au meilleur modèle de ce type en particulier. Si vous ne spécifiez pas ce paramètre, un type de problème est détecté pendant l'entraînement, en fonction de vos données.

OBJECTIVE ('MSE' | 'Accuracy' | 'F1' | 'F1Macro' | 'AUC')

(Facultatif) Spécifie le nom de la métrique d'objectif utilisée pour mesurer la qualité prédictive d'un système de machine learning. Cette métrique est optimisée pendant l'entraînement afin de fournir la meilleure estimation des valeurs des paramètres du modèle à partir des données. Si vous ne spécifiez pas de métrique explicitement, le comportement par défaut consiste à utiliser automatiquement MSE : pour la régression, F1 : pour la classification binaire, Précision : pour la classification multiclasse. Pour plus d'informations sur les objectifs, consultez Auto MLJob Objective dans le manuel Amazon SageMaker AI API Reference.

MAX_CELLS integer

(Facultatif) Spécifie le nombre de cellules dans les données d'entraînement. Cette valeur est le produit du nombre d'enregistrements (dans la requête d'entraînement ou dans la table) multiplié par le nombre de colonnes. La valeur par défaut est 1 000 000.

MAX_RUNTIME integer

(Facultatif) Spécifie la durée maximale d'entraînement. Les tâches d'entraînement se terminent souvent plus tôt en fonction de la taille du jeu de données. Ce nombre spécifie la durée maximale de l'entraînement. La valeur par défaut est 5 400 (90 minutes).

S3_GARBAGE_COLLECT { ON | OFF }{SUR | OFF}

(Facultatif) Spécifie si Amazon Redshift lance le récupérateur de mémoire sur les jeux de données obtenus utilisés pour entraîner les modèles et sur les modèles. Si la valeur est définie sur OFF, les jeux de données obtenus utilisés pour entraîner les modèles et les modèles eux-mêmes restent dans Amazon S3 et peuvent être utilisés à d'autres fins. Si la valeur est définie sur ON, Amazon Redshift supprime les artefacts dans Amazon S3 une fois l'entraînement terminé. La valeur par défaut est ON.

KMS_KEY_ID 'kms_key_id'

(Facultatif) Spécifie si Amazon Redshift utilise le chiffrement côté serveur avec une clé AWS KMS pour protéger les données au repos. Les données en transit sont protégées par le protocole SSL.

PREPROCESSORS ’string’

(Facultatif) Spécifie certaines combinaisons de préprocesseurs à certains ensembles de colonnes. Le format est une liste de columnSets et les transformations appropriées à appliquer à chaque ensemble de colonnes. Amazon Redshift applique tous les transformateurs d'une liste de transformateurs spécifique à toutes les colonnes de la liste correspondante. ColumnSet Par exemple, pour appliquer OneHotEncoder avec Imputer aux colonnes t1 et t2, utilisez l'exemple de commande suivant.

```
CREATE MODEL customer_churn
FROM customer_data
TARGET 'Churn'
FUNCTION predict_churn
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
PROBLEM_TYPE BINARY_CLASSIFICATION
OBJECTIVE 'F1'
PREPROCESSORS '[
...
{"ColumnSet": [
    "t1",
    "t2"
  ],
  "Transformers": [
    "OneHotEncoder",
    "Imputer"
  ]
},
{"ColumnSet": [
    "t3"
  ],
  "Transformers": [
    "OneHotEncoder"
  ]
},
{"ColumnSet": [
    "temp"
  ],
  "Transformers": [
    "Imputer",
    "NumericPassthrough"
  ]
}
]'
```

```
SETTINGS (
S3_BUCKET 'amzn-s3-demo-bucket'
)
```

Amazon Redshift prend en charge les transformateurs suivants :

- OneHotEncoder — Généralement utilisé pour coder une valeur discrète dans un vecteur binaire avec une valeur différente de zéro. Ce transformateur convient à de nombreux modèles de machine learning.
- OrdinalEncoder — Encode des valeurs discrètes en un seul entier. Ce transformateur convient à certains modèles de machine learning, tels que MLP et Linear Learner.
- NumericPassthrough — Transmet l'entrée telle quelle dans le modèle.
- Imputer : remplit les valeurs manquantes et les valeurs qui ne sont pas un nombre (NaN).
- ImputerWithIndicator — Complète les valeurs manquantes et les valeurs NaN. Ce transformateur crée également un indicateur indiquant si des valeurs étaient manquantes et ont été emplies.
- Normalizer : normalise les valeurs, ce qui peut améliorer les performances de nombreux algorithmes de machine learning.
- DateTimeVectorizer — Crée une intégration vectorielle, représentant une colonne de type de données date/heure pouvant être utilisée dans les modèles d'apprentissage automatique.
- PCA : projette les données dans un espace dimensionnel inférieur afin de réduire le nombre de fonctions tout en conservant autant d’informations que possible.
- StandardScaler — Normalise les caractéristiques en supprimant la moyenne et en les adaptant à l'unité de variance.
- MinMax — Transforme les entités en adaptant chaque entité à une plage donnée.

Amazon Redshift ML stocke les transformateurs entraînés et les applique automatiquement dans le cadre de la requête de prédiction. Vous n’avez pas besoin de les spécifier lorsque vous générez des prédictions à partir de votre modèle.

CRÉEZ des XGBoost modèles avec AUTO OFF

En général, la commande AUTO OFF CREATE MODEL a des objectifs différents de ceux de la commande CREATE MODE par défaut.

Si vous êtes utilisateur avancé qui connaît déjà le type de modèle souhaité et les hyperparamètres à utiliser lors de l’entraînement de ces modèles, vous pouvez utiliser CREATE MODEL avec

AUTO OFF pour désactiver la détection automatique CREATE MODEL des préprocesseurs et des hyperparamètres. Pour ce faire, vous devez spécifier explicitement le type de modèle. XGBoost est actuellement le seul type de modèle pris en charge lorsque AUTO est réglé sur OFF. Vous pouvez spécifier des hyperparamètres. Amazon Redshift utilise des valeurs par défaut pour tous les hyperparamètres que vous avez spécifiés.

CRÉEZ des XGBoost modèles avec la syntaxe AUTO OFF

```
CREATE MODEL model_name
FROM { table_name | (select_statement ) }
TARGET column_name
FUNCTION function_name
IAM_ROLE { default }
AUTO OFF
MODEL_TYPE XGBOOST
OBJECTIVE { 'reg:squarederror' | 'reg:squaredlogerror' | 'reg:logistic' |
            'reg:pseudohubererror' | 'reg:tweedie' | 'binary:logistic' | 'binary:hinge'
 |
            'multi:softmax' | 'rank:pairwise' | 'rank:ndcg' }
HYPERPARAMETERS DEFAULT EXCEPT (
    NUM_ROUND '10',
    ETA '0.2',
    NUM_CLASS '10',
    (, ...)
)
PREPROCESSORS 'none'
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket', |
  S3_GARBAGE_COLLECT { ON | OFF }, |
  KMS_KEY_ID 'kms_key_id', |
  MAX_CELLS integer, |
  MAX_RUNTIME integer (, ...)
)
```

CRÉEZ des XGBoost modèles avec les paramètres AUTO OFF

AUTO OFF

Désactive la détection automatique CREATE MODEL du préprocesseur, de l'algorithme et de la sélection d'hyperparamètres.

MODEL_TYPE XGBOOST

Indiquer d’utiliser XGBOOST pour entraîner le modèle.

OBJECTIVE str

Spécifie un objectif reconnu par l’algorithme. Amazon Redshift prend en charge reg:squarederror, reg:squaredlogerror, reg:logistic, reg:pseudohubererror, reg:tweedie, binary:logistic, binary:hinge, multi:softmax. Pour plus d'informations sur ces objectifs, consultez la section Paramètres des tâches d'apprentissage dans la XGBoost documentation.

HYPERPARAMETERS { DEFAULT | DEFAULT EXCEPT (key ‘value’ (,..)) }

Spécifie si les XGBoost paramètres par défaut sont utilisés ou remplacés par des valeurs spécifiées par l'utilisateur. Les valeurs doivent être entre guillemets simples. Vous trouverez ci-dessous des exemples de paramètres pour XGBoost et leurs valeurs par défaut.

Nom du paramètre	Valeur de paramètre	Valeur par défaut	Remarques
num_class	Entier	Requis pour la classification multiclasse.	N/A
num_round	Entier	100	N/A
tree_method	Chaîne	Auto	N/A
max_depth	Entier	6	[0 , 10]
min_child_weight	Float	1	MinValue: 0, MaxValue 120
subsample	Float	1	MinValue: 0,5, MaxValue : 1

Nom du paramètre	Valeur de paramètre	Valeur par défaut	Remarques
gamma	Float	0	MinValue: 0, MaxValue 5
alpha	Float	0	MinValue: 0, MaxValue 100
eta	Float	0.3	MinValue: 0,1, MaxValue 0,5
colsample_byleve	Float	1	MinValue: 0,1, MaxValue : 1
colsample_bynode	Float	1	MinValue: 0,1, MaxValue : 1
colsample_bytree	Float	1	MinValue: 0,5, MaxValue : 1
lambda	Float	1	MinValue: 0, MaxValue 100
max_delta_step	Entier	0	[0, 10]

L'exemple suivant prépare les données pour XGBoost.

```
DROP TABLE IF EXISTS abalone_xgb;

CREATE TABLE abalone_xgb (
length_val float,
diameter float,
height float,
whole_weight float,
shucked_weight float,
viscera_weight float,
shell_weight float,
rings int,
record_number int);

COPY abalone_xgb
```

```
FROM 's3://redshift-downloads/redshift-ml/abalone_xg/'
REGION 'us-east-1'
IAM_ROLE default
IGNOREHEADER 1 CSV;
```

L'exemple suivant crée un XGBoost modèle avec des options avancées spécifiées, telles que MODEL_TYPE, OBJECTIVE et PREPROCESSORS.

```
DROP MODEL abalone_xgboost_multi_predict_age;

CREATE MODEL abalone_xgboost_multi_predict_age
FROM ( SELECT length_val,
              diameter,
              height,
              whole_weight,
              shucked_weight,
              viscera_weight,
              shell_weight,
              rings
   FROM abalone_xgb WHERE record_number < 2500 )
TARGET rings FUNCTION ml_fn_abalone_xgboost_multi_predict_age
IAM_ROLE default
AUTO OFF
MODEL_TYPE XGBOOST
OBJECTIVE 'multi:softmax'
PREPROCESSORS 'none'
HYPERPARAMETERS DEFAULT EXCEPT (NUM_ROUND '100', NUM_CLASS '30')
SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket');
```

L’exemple suivant utilise une requête d’inférence pour prédire l’âge du poisson dont le nombre d’enregistrements est supérieur à 2 500. Il utilise la fonction ml_fn_abalone_xgboost_multi_predict_age créée à partir de la commande ci-dessus.

```
select ml_fn_abalone_xgboost_multi_predict_age(length_val,
                                                diameter,
                                                height,
                                                whole_weight,
                                                shucked_weight,
                                                viscera_weight,
                                                shell_weight)+1.5 as age
from abalone_xgb where record_number > 2500;
```

Modèle BYOM (Bring Your Own Model) : inférence locale

Amazon Redshift ML prend en charge l'utilisation du modèle BYOM pour l'inférence locale.

Vous trouverez ci-dessous un résumé des options de syntaxe CREATE MODEL pour le modèle BYOM. Vous pouvez utiliser un modèle formé en dehors d'Amazon Redshift avec Amazon SageMaker AI pour effectuer des inférences dans la base de données localement dans Amazon Redshift.

Syntaxe CREATE MODEL pour l'inférence locale

Voici une description de la syntaxe CREATE MODEL pour l'inférence locale.

```
CREATE MODEL model_name
FROM ('job_name' | 's3_path' )
FUNCTION function_name ( data_type [, ...] )
RETURNS data_type
IAM_ROLE { default }
[ SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket', | --required
  KMS_KEY_ID 'kms_string') --optional
];
```

Amazon Redshift ne prend actuellement en charge que les modèles préentraînés XGBoost, MLP et Linear Learner pour le BYOM. Vous pouvez importer le pilote automatique SageMaker AI et les modèles directement entraînés dans Amazon SageMaker AI à des fins d'inférence locale en utilisant ce chemin.

Paramètres CREATE MODEL pour l'inférence locale

model_name

Nom du modèle. Le nom du modèle d'un schéma doit être unique.

FROM ('job_name' | 's3_path')

Le job_name utilise un nom de tâche Amazon SageMaker AI comme entrée. Le nom du poste peut être un nom de tâche de formation Amazon SageMaker AI ou un nom de tâche Amazon SageMaker AI Autopilot. La tâche doit être créée dans le compte AWS propriétaire du cluster Amazon Redshift. Pour trouver le nom du poste, lancez Amazon SageMaker AI. Dans le menu déroulant Training (Entraînement), choisissez Training jobs (Tâches d'entraînement).

’s3_path’ spécifie l’emplacement S3 du fichier d’artefacts de modèle .tar.gz à utiliser lors de la création du modèle.

FUNCTION function_name (data_type [, ...])

Nom de la fonction à créer et types de données des arguments d’entrée. Vous pouvez spécifier un nom de schéma.

RETURNS type_données

Type de données de la valeur renvoyée par la fonction.

IAM_ROLE { default | ’arn:aws:iam::<account-id>:role/<role-name>’}

Utilisez le mot clé par défaut pour qu’Amazon Redshift utilise le rôle IAM défini comme rôle par défaut et associé au cluster lorsque la commande CREATE MODEL s’exécute.

Utilisez l’Amazon Resource Name (ARN) d’un rôle IAM que votre cluster utilise pour l’authentification et l’autorisation.

PARAMÈTRES (S3_BUCKET 'amzn-s3-demo-bucket', | KMS_KEY_ID 'kms_string')

La clause S3_BUCKET spécifie l’emplacement Amazon S3 utilisé pour stocker les résultats intermédiaires.

(Facultatif) La clause KMS_KEY_ID indique si Amazon Redshift utilise le chiffrement côté serveur avec une clé pour protéger les données au repos. AWS KMS Les données en transit sont protégées par le protocole SSL.

Pour plus d'informations, consultez CREATE MODEL avec guide de l’utilisateur.

Exemple CREATE MODEL pour l’inférence locale

L'exemple suivant crée un modèle qui a déjà été formé à Amazon SageMaker AI, en dehors d'Amazon Redshift. Étant donné que le type de modèle est pris en charge par le ML d’Amazon Redshift pour l’inférence locale, la commande CREATE MODEL suivante crée une fonction qui peut être utilisée localement dans Amazon Redshift. Vous pouvez fournir un nom de poste de formation à l' SageMaker IA.

```
CREATE MODEL customer_churn
FROM 'training-job-customer-churn-v4'
FUNCTION customer_churn_predict (varchar, int, float, float)
RETURNS int
IAM_ROLE default
```

```
SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket');
```

Une fois le modèle créé, vous pouvez utiliser la fonction customer_churn_predict avec les types d'arguments spécifiés pour effectuer des prédictions.

Modèle BYOM (Bring Your Own Model) : inférence distante

Amazon Redshift ML prend également en charge l'utilisation du modèle BYOM pour l'inférence distante.

Vous trouverez ci-dessous un résumé des options de syntaxe CREATE MODEL pour le modèle BYOM.

Syntaxe CREATE MODEL pour l'inférence distante

Voici une description de la syntaxe CREATE MODEL pour l'inférence distante.

```
CREATE MODEL model_name
FUNCTION function_name ( data_type [, ...] )
RETURNS data_type
SAGEMAKER 'endpoint_name'['model_name']
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
[SETTINGS (MAX_BATCH_ROWS integer)];
```

Paramètres CREATE MODEL pour l'inférence distante

model_name

Nom du modèle. Le nom du modèle d'un schéma doit être unique.

FUNCTION fn_name ([data_type] [, ...])

Nom de la fonction et types de données des arguments d'entrée. Consultez Types de données pour connaître tous les types de données pris en charge. `Geography`, `geometry` et `hllsketch` ne sont pas pris en charge.

Vous pouvez également fournir un nom de fonction dans un schéma en utilisant une notation en deux parties, telle que`myschema.myfunction`.

RETURNS type_données

Type de données de la valeur renvoyée par la fonction. Consultez Types de données pour connaître tous les types de données pris en charge. `Geography`, `geometry` et `hllsketch` ne sont pas pris en charge.

SAGEMAKER ’endpoint_name’[:’model_name’]

Nom du point de terminaison Amazon SageMaker AI. Si le nom du point de terminaison pointe vers un point de terminaison multimodèle, ajoutez le nom du modèle à utiliser. Le point de terminaison doit être hébergé dans la même AWS région et Compte AWS que le cluster Amazon Redshift. Pour trouver votre point de terminaison, lancez Amazon SageMaker AI. Dans le menu déroulant Inference (Inférence), choisissez Endpoints (Points de terminaison).

IAM_ROLE { default | ’arn:aws:iam::<account-id>:role/<role-name>’}

Utilisez le mot clé par défaut pour qu’Amazon Redshift utilise le rôle IAM défini comme rôle par défaut et associé au cluster lorsque la commande CREATE MODEL s’exécute. Vous pouvez également spécifier l’ARN d’un rôle IAM pour utiliser ce rôle.

MAX_BATCH_ROWS entier

Le nombre maximum de lignes qu'Amazon Redshift envoie dans une seule demande par lot pour un seul appel d' SageMaker IA. Il n'est pris en charge que pour le BYOM avec inférence à distance. Le nombre réel de lignes dans un lot dépend également de la taille d'entrée, mais il est inférieur ou égal à cette valeur. La valeur minimale pour ce paramètre est 1. La valeur maximale est `INT_MAX` de 2 147 483 647. Ce paramètre est obligatoire uniquement lorsque les types de données en entrée et en sortie le sont`SUPER`. La valeur par défaut est`INT_MAX`, ou 2 147 483 647.

Lorsque le modèle est déployé sur un SageMaker point de terminaison d' SageMaker IA, l'IA crée les informations du modèle dans Amazon Redshift. Il effectue ensuite une inférence à travers la fonction externe. Vous pouvez utiliser la commande SHOW MODEL pour afficher les informations de modèle sur votre cluster Amazon Redshift.

CREATE MODEL pour les notes d’utilisation d’inférence distante

Avant d’utiliser CREATE MODLE pour l’inférence distante, tenez compte des éléments suivants :

- Le point de terminaison doit être hébergé par le même AWS compte que celui qui possède le cluster Amazon Redshift.
- Assurez-vous que le point de terminaison Amazon SageMaker AI dispose de suffisamment de ressources pour prendre en charge les appels d'inférence d'Amazon Redshift ou que le point de terminaison SageMaker Amazon AI peut être automatiquement redimensionné.

- Si vous n'utilisez pas le type de SUPER données comme entrée, le modèle accepte uniquement les entrées au format de valeurs séparées par des virgules (CSV), ce qui correspond à un type de contenu `text/CSV` dans SageMaker AI.
- Si vous n'utilisez pas le type de SUPER données en entrée, la sortie des modèles est une valeur unique du type spécifié lors de la création de la fonction. La sortie est au format de valeurs séparées par des virgules (CSV) via un type de `text/CSV` contenu AI. SageMaker `VARCHAR`les types de données ne peuvent pas être entre guillemets et ne peuvent pas contenir de nouvelles lignes, et chaque sortie doit figurer sur une nouvelle ligne.
- Les modèles acceptent les valeurs NULL comme chaînes vides.
- Lorsque le type de données d'entrée estSUPER, un seul argument d'entrée est pris en charge.
- Lorsque le type de données d'entrée estSUPER, le type de données renvoyé doit également l'êtreSUPER.
- MAX_BATCH_ROWS est requis lorsque les types de données en entrée et en sortie sont de type SUPER.
- Lorsque le type de données d'entrée est SUPER le type de contenu du point de terminaison, c'est soit `application/json` lorsque MAX_BATCH_ROWS l'est1, soit dans tous les autres cas. `application/jsonlines`
- Lorsque le type de données renvoyé est SUPER le type d'acceptation, l'invocation du point de terminaison est soit `application/json` lorsque MAX_BATCH_ROWS l'est1, soit dans tous les autres cas. `application/jsonlines`

Exemple de CREATE MODEL pour l'inférence distante

L'exemple suivant crée un modèle qui utilise un point de terminaison d' SageMaker IA pour effectuer des prédictions. Assurez-vous que le point de terminaison est en cours d’exécution pour effectuer des prédictions et spécifiez son nom dans la commande CREATE MODEL.

```
CREATE MODEL remote_customer_churn
FUNCTION remote_fn_customer_churn_predict (varchar, int, float, float)
RETURNS int
SAGEMAKER 'customer-churn-endpoint'
IAM_ROLE default;
```

L'exemple suivant crée un BYOM avec inférence à distance avec un grand modèle de langage (LLM). LLMs hébergés sur Amazon SageMaker AI Jumpstart, ils acceptent et renvoient le type de `application/json` contenu et ne prennent en charge qu'un seul JSON par appel. Les types de

données d'entrée et de retour doivent être définis sur 1 SUPER et MAX_BATCH_ROWS doit être défini sur 1.

```
CREATE MODEL sample_super_data_model
FUNCTION sample_super_data_model_predict(super)
RETURNS super
SAGEMAKER 'sample_super_data_model_endpoint'
IAM_ROLE default
SETTINGS (MAX_BATCH_ROWS 1);
```

CREATE MODEL avec K-MEANS

Amazon Redshift prend en charge l'algorithme K-Means qui regroupe des données non étiquetées. Cet algorithme résout les problèmes de mise en cluster lorsque vous souhaitez découvrir des regroupements dans les données. Les données non classifiées sont regroupées et partitionnées en fonction de leurs similitudes et de leurs différences.

CREATE MODEL avec la syntaxe K-MEANS

```
CREATE MODEL model_name
FROM { table_name | ( select_statement ) }
FUNCTION function_name
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }
AUTO OFF
MODEL_TYPE KMEANS
PREPROCESSORS 'string'
HYPERPARAMETERS DEFAULT EXCEPT ( K 'val' [, ...] )
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket',
  KMS_KEY_ID 'kms_string', |
    -- optional
  S3_GARBAGE_COLLECT on / off, |
    -- optional
  MAX_CELLS integer, |
    -- optional
  MAX_RUNTIME integer
    -- optional);
```

CREATE MODEL avec les paramètres K-MEANS

AUTO OFF

Désactive la détection automatique CREATE MODEL du préprocesseur, de l'algorithme et de la sélection d'hyperparamètres.

MODEL_TYPE KMEANS

Permet de spécifier l'utilisation de KMEANS pour entraîner le modèle.

PREPROCESSORS 'string'

Spécifie certaines combinaisons de préprocesseurs à certains ensembles de colonnes. Le format est une liste de columnSets et les transformations appropriées à appliquer à chaque ensemble de colonnes. Amazon Redshift prend en charge 3 préprocesseurs K-Means, à savoir StandardScaler, et. MinMax NumericPassthrough Si vous ne souhaitez appliquer aucun prétraitement pour K-Means, choisissez-le NumericPassthrough explicitement en tant que transformateur. Pour plus d'informations sur les transformateurs pris en charge, consultez CREATE MODEL avec paramètres de guide de l'utilisateur.

L'algorithme K-Means utilise la distance euclidienne pour calculer la similarité. Le prétraitement des données garantit que les fonctions du modèle restent à la même échelle et produisent des résultats fiables.

HYPERPARAMETERS DEFAULT EXCEPT (K 'val' [, ...])

Spécifie si les paramètres K-Means sont utilisés. Vous devez spécifier le paramètre K lors de l'utilisation de l'algorithme K-Means. Pour plus d'informations, consultez K-Means Hyperparameters dans le manuel Amazon SageMaker AI Developer Guide

L'exemple suivant prépare des données pour K-Means.

```
CREATE MODEL customers_clusters
FROM customers
FUNCTION customers_cluster
IAM_ROLE default
AUTO OFF
MODEL_TYPE KMEANS
PREPROCESSORS '[
{
  "ColumnSet": [ "*" ],
```

```
  "Transformers": [ "NumericPassthrough" ]
}
]'
HYPERPARAMETERS DEFAULT EXCEPT ( K '5' )
SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket');

select customer_id, customers_cluster(...) from customers;
customer_id | customers_cluster
--------------------
12345            1
12346            2
12347            4
12348
```

CREATE MODEL avec Forecast

Les modèles de prévisions de Redshift ML utilisent Amazon Forecast pour créer des prévisions chronologiques précises. Cela vous permet d'utiliser des données historiques sur une période donnée pour faire des prédictions sur des événements futurs. Les cas d'utilisation courants d'Amazon Forecast incluent l'utilisation des données sur les produits vendus au détail pour déterminer le prix des stocks, les données de quantité de fabrication pour prédire la quantité à commander d'un article et les données de trafic Web pour prévoir le volume de trafic qu'un serveur Web est susceptible de recevoir.

Les limites de quota d'Amazon Forecast sont appliquées dans les modèles de prévisions Amazon Redshift. Par exemple, le nombre maximum de prévisions est de 100, mais il est ajustable. La suppression d'un modèle de prévision ne supprime pas automatiquement les ressources associées dans Amazon Forecast. Si vous supprimez un cluster Redshift, tous les modèles associés sont également supprimés.

Notez que les modèles de prévisions ne sont actuellement disponibles que dans les régions suivantes :

- USA Est (Ohio) (us-east-2)
- USA Est (Virginie du Nord) (us-east-1)
- USA Ouest (Oregon) (us-west-2)
- Asie-Pacifique (Mumbai) (ap-south-1)
- Asie-Pacifique (Séoul) (ap-northeast-2)
- Asie-Pacifique (Singapour) (ap-southeast-1)

- Asie-Pacifique (Sydney) (ap-southeast-2)
- Asie-Pacifique (Tokyo) (ap-northeast-1)
- Europe (Francfort) (eu-central-1)
- Europe (Irlande) (eu-west-1)

CREATE MODEL avec une syntaxe Forecast

```
CREATE [ OR REPLACE ] MODEL forecast_model_name
FROM { table_name | ( select_query ) }
TARGET column_name
IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>'}
AUTO ON
MODEL_TYPE FORECAST
SETTINGS (
  S3_BUCKET 'amzn-s3-demo-bucket',
  HORIZON integer,
  FREQUENCY forecast_frequency
  [PERCENTILES '0.1', '0.5', '0.9']
  )
```

CREATE MODEL avec des paramètres Forecast

forecast_model_name

Nom du modèle. Le nom du modèle doit être unique.

FROM { table_name | (select_query) }

Le table_name ou la requête qui spécifie les données d'entraînement. Il peut s'agir d'une table existante dans le système ou d'une requête SELECT compatible avec Amazon RedShift, placée entre parenthèses. Le résultat de la table ou de la requête doit comporter au moins trois colonnes : (1) une colonne varchar qui indique le nom de la série chronologique. Chaque jeu de données peut comporter plusieurs séries chronologiques ; (2) une colonne date/heure ; et (3) la colonne cible pour les prévisions. Cette colonne cible doit être un nombre entier ou à virgule flottante. Si vous fournissez un jeu de données comportant plus de trois colonnes, Amazon Redshift part du principe que toutes les colonnes supplémentaires font partie d'une série chronologique associée. Notez que les séries chronologiques associées doivent être de type nombre entier ou à virgule flottante. Pour plus d'informations sur les séries chronologiques associées, consultez Utilisation de jeux de données de séries chronologiques associés.

TARGET column_name

Nom de la colonne qui devient la cible de la prédiction. La colonne doit exister dans la clause FROM.

IAM_ROLE { default | 'arn:aws:iam::<account-id>:role/<role-name>' }

Utilisez le mot clé par défaut pour qu'Amazon Redshift utilise le rôle IAM défini comme rôle par défaut et associé au cluster lorsque la commande CREATE MODEL s'exécute. Vous pouvez également spécifier l'ARN d'un rôle IAM pour utiliser ce rôle.

AUTO ON

Active la détection automatique CREATE MODEL de l'algorithme et de la sélection d'hyperparamètres. Si vous spécifiez « on » lors de la création d'un modèle de prévision AutoPredictor, vous devez utiliser un modèle Forecast, dans lequel Amazon Forecast applique les combinaisons optimales d'algorithmes à chaque série chronologique de votre ensemble de données.

MODEL_TYPE FORECAST

Permet de spécifier l'utilisation de FORECAST pour entraîner le modèle.

S3_BUCKET 'amzn-s3-demo-bucket'

Nom du compartiment Amazon Simple Storage Service que vous avez créé précédemment et qui est utilisé pour partager des données d'entraînement et des artefacts entre Amazon Redshift et Amazon Forecast. Amazon Redshift crée un sous-dossier dans ce compartiment avant de décharger les données d'entraînement. Lorsque l'entraînement est terminé, Amazon Redshift supprime le sous-dossier créé et son contenu.

Entier HORIZON

Le nombre maximum de prédictions que le modèle de prévision peut renvoyer. Une fois le modèle entraîné, vous ne pouvez pas modifier cet entier.

FREQUENCY forecast_frequency

Indique le degré de granularité que vous souhaitez pour les prévisions. Les options disponibles sont `Y | M | W | D | H | 30min | 15min | 10min | 5min | 1min`. Obligatoire si vous entraînez un modèle de prévision.

Chaîne PERCENTILES

Une chaîne délimitée par des virgules qui spécifie les types de prévisions utilisés pour entraîner un prédicteur. Les types de prévisions peuvent être des quantiles compris entre 0,01 et 0,99, par

incréments de 0,01 ou plus. Vous pouvez également spécifier la prévision moyenne à l'aide de la moyenne. Vous pouvez spécifier un maximum de cinq types de prévisions.

L'exemple suivant montre comment créer un modèle de prévision simple.

```
CREATE MODEL forecast_example
FROM forecast_electricity_
TARGET target
IAM_ROLE 'arn:aws:iam::<account-id>:role/<role-name>'
AUTO ON
MODEL_TYPE FORECAST
SETTINGS (S3_BUCKET 'amzn-s3-demo-bucket',
          HORIZON 24,
          FREQUENCY 'H',
          PERCENTILES '0.25,0.50,0.75,mean',
          S3_GARBAGE_COLLECT OFF);
```

Après avoir créé le modèle de prévision, vous pouvez créer une nouvelle table contenant les données de prédiction.

```
CREATE TABLE forecast_model_results as SELECT Forecast(forecast_example)
```

Vous pouvez ensuite interroger la nouvelle table pour obtenir des prédictions.

```
SELECT * FROM forecast_model_results
```

CREATE PROCEDURE

Crée une nouvelle procédure stockée ou remplace une procédure existante pour la base de données actuelle.

Pour plus d'informations et d'exemples, consultez Création de procédures stockées dans Amazon Redshift.

Privilèges requis

Vous devez être autorisé par l'une des méthodes suivantes pour exécuter la procédure CREATE OR REPLACE :

- Pour CREATE PROCEDURE :

- Superuser
- Utilisateurs disposant des privilèges CREATE et USAGE sur le schéma dans lequel la procédure stockée est créée

- Pour REPLACE PROCEDURE :
  - Superuser
  - Propriétaire de la procédure

Syntaxe

```
CREATE [ OR REPLACE ] PROCEDURE sp_procedure_name
  ( [ [ argname ] [ argmode ] argtype [, ...] ] )
[ NONATOMIC ]
AS $$
  procedure_body
$$ LANGUAGE plpgsql
[ { SECURITY INVOKER | SECURITY DEFINER } ]
[ SET configuration_parameter { TO value | = value } ]
```

Paramètres

OR REPLACE

Clause qui spécifie que si une procédure ayant le même nom et les mêmes types de données, ou signature, existe déjà, la procédure existante est remplacée. Vous pouvez uniquement remplacer une procédure par une nouvelle procédure qui définit un ensemble identique de types de données.

Si vous définissez une procédure avec le même nom qu'une procédure existante, mais une signature différente, vous créez une nouvelle procédure. Autrement dit, le nom de la procédure est surchargé. Pour plus d'informations, consultez Surcharge des noms de procédure.

sp_procedure_name

Nom de la procédure. Si vous spécifiez un nom de schéma (tel que **`myschema.myprocedure`**), la procédure est créée à dans le schéma spécifié. Sinon, la procédure est créée dans le schéma en cours. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

Nous vous conseillons d'utiliser le préfixe `sp_` pour tous les noms des procédures stockées. Amazon Redshift réserve le préfixe `sp_` pour les noms de procédures stockées. Utiliser le préfixe

`sp_` vous permet de garantir que le nom de votre procédure stockée n'est en conflit avec aucun nom de fonction ou de procédure stockée intégré à Amazon Redshift existant ou futur. Pour plus d'informations, consultez Dénomination des procédures stockées.

Vous pouvez définir plusieurs procédures portant le même nom si les types de données des arguments en entrée, ou signature, sont différents. Autrement dit, dans ce cas, le nom de la procédure est surchargé. Pour plus d'informations, consultez Surcharge des noms de procédure

[argname] [argmode] argtype

Liste de noms d'arguments, de modes d'argument et de types de données. Seul le type de données est requis. Le nom et le mode sont facultatifs, et leur position peut être inversée.

Le mode de l'argument peut être IN, OUT ou INOUT. La valeur par défaut est IN.

Vous pouvez utiliser des arguments OUT et INOUT pour renvoyer une ou plusieurs valeurs à partir d'un appel de procédure. S'il existe des arguments OUT ou INOUT, l'appel de procédure renvoie une ligne de résultats contenant n colonnes, où n est le nombre total d'arguments OUT ou INOUT.

Les arguments INOUT sont des arguments à la fois en entrée et en sortie. Les arguments en entrée comprennent les arguments IN et INOUT, et les arguments en sortie comprennent les arguments OUT et INOUT.

Les arguments OUT ne sont pas spécifiés dans le cadre de l'instruction CALL. Spécifiez les arguments INOUT dans l'instruction CALL de la procédure stockée. Les arguments INOUT peuvent s'avérer utiles lorsque vous transmettez ou renvoyez des valeurs à partir d'appels imbriqués, ainsi que lors du renvoi d'un type `refcursor`. Pour plus d'informations sur les types `refcursor`, consultez Curseurs.

Les types de données d'arguments peuvent être de n'importe quel type de données Amazon Redshift standard. En outre, `refcursor` peut être un type de données d'argument.

Vous pouvez spécifier un maximum de 32 arguments en entrée et de 32 arguments en sortie.

AS $$ procedure_body $$

Construction qui englobe la procédure à exécuter. Les mots-clés littéraux AS $$ et $$ sont obligatoires.

Amazon Redshift requiert que vous placiez l'instruction dans votre procédure à l'aide d'un format appelé guillemets dollar. Tout ce qui se trouve dans l'encadrement est passé exactement comme

tel. Vous n'avez pas besoin de définir de séquence d'échappement pour les caractères spéciaux, car les contenus de la chaîne sont écrits littéralement.

Avec les guillemets dollar, vous utilisez une paire de symboles dollar ($$) pour marquer le début et la fin de l'instruction à exécuter, comme illustré dans l'exemple suivant.

```
$$ my statement $$
```

Le cas échéant, entre les symboles dollar de chaque paire, vous pouvez spécifier une chaîne pour aider à identifier l'instruction. La chaîne que vous utilisez doit être la même au début et à la fin des paires d'encadrement. Cette chaîne est sensible à la casse et suit les mêmes contraintes qu'un identificateur sans guillemets, sauf que celui-ci ne peut pas contenir de symboles dollar. L'exemple suivant utilise la chaîne test.

```
$test$ my statement $test$
```

Cette syntaxe est également utile les guillemets dollar imbriqués. Pour plus d'informations sur les guillemets dollar, consultez relative aux constantes de chaîne avec guillemet dollar dans Lexical Structure dans la documentation PostgreSQL.

procedure_body

Un ensemble de PL/pgSQL déclarations valides. PL/pgSQL les instructions ajoutent aux commandes SQL des structures procédurales, notamment des expressions conditionnelles et des boucles, afin de contrôler le flux logique. La plupart des commandes SQL peuvent être utilisées dans le corps de la procédure, y compris celles du langage de modification de données (DML) telles que COPY, UNLOAD et INSERT, et celles du langage de définition de données (DDL) telles que CREATE TABLE. Pour plus d'informations, consultez Guide de référence du langage PL/pgSQL.

LANGUAGE plpgsql

Valeur pour le langage. Spécifiez `plpgsql`. Vous devez avoir l'autorisation pour USAGE ON LANGUAGE pour utiliser `plpgsql`. Pour de plus amples informations, veuillez consulter GRANT.

NONATOMIC

Crée la procédure stockée dans un mode de transaction non atomique. Le mode NONATOMIC valide automatiquement les déclarations à l'intérieur de la procédure. En outre, lorsqu'une erreur se produit à l'intérieur de la procédure NONATOMIC, l'erreur n'est pas relancée si elle est gérée par un bloc d'exception. Pour plus d'informations, consultez Gestion des transactions et RAISE.

Lorsque vous définissez une procédure stockée comme NONATOMIC, tenez compte des éléments suivants :

- Lorsque vous imbriquez des appels de procédures stockées, toutes les procédures doivent être créées dans le même mode de transaction.
- L'option SECURITY DEFINER et l'option SET configuration_parameter ne sont pas prises en charge lors de la création d'une procédure en mode NONATOMIC.
- Tout curseur ouvert (explicitement ou implicitement) est fermé automatiquement lorsqu'une validation implicite est effectuée. Par conséquent, vous devez ouvrir une transaction explicite avant de commencer une boucle de curseur pour vous assurer que tout code SQL dans l'itération de la boucle n'est pas implicitement validé.

SECURITY INVOKER | SECURITY DEFINER

L'option SECURITY DEFINER n'est pas prise en charge lorsque NONATOMIC est spécifié.

Le mode de sécurité pour la procédure détermine les privilèges d'accès de la procédure lors de l'exécution. La procédure doit être autorisée à accéder aux objets de la base de données sous-jacente.

Pour le mode SECURITY INVOKER, la procédure utilise les privilèges de l'utilisateur appelant la procédure. L'utilisateur doit disposer d'autorisations explicites sur les objets de la base de données sous-jacente. La valeur par défaut est SECURITY INVOKER.

Pour le mode SECURITY DEFINER, la procédure utilise les privilèges du propriétaire de la procédure. Le propriétaire de la procédure est défini comme l'utilisateur propriétaire de la procédure au moment de l'exécution, pas nécessairement l'utilisateur qui a initialement défini la procédure. L'utilisateur qui appelle la procédure doit détenir un privilège EXECUTE sur la procédure, mais il n'a pas besoin de privilèges sur les objets sous-jacents.

SET configuration_parameter { TO value | = value }

Ces options ne sont pas prises en charge lorsque NONATOMIC est spécifié.

La clause SET entraîne la définition du configuration_parameter spécifié sur la valeur spécifiée au lancement de la procédure. Cette clause restaure ensuite configuration_parameter à sa valeur précédente à la fin de la procédure.

Notes d'utilisation

Si une procédure stockée a été créée à l'aide de l'option SECURITY DEFINER, lors de l'appel de la fonction CURRENT_USER depuis la procédure stockée, Amazon Redshift renvoie le nom d'utilisateur du propriétaire de la procédure stockée.

Exemples

> **Note**
>
> Si vous rencontrez une erreur similaire à ce qui suit lors de l'exécution de ces exemples :
>
> ```
> ERROR: 42601: [Amazon](500310) unterminated dollar-quoted string at or near "$$
> ```
>
> Consultez Présentation des procédures stockées dans Amazon Redshift.

L'exemple suivant crée une procédure avec deux paramètres d'entrée.

```
CREATE OR REPLACE PROCEDURE test_sp1(f1 int, f2 varchar(20))
AS $$
DECLARE
  min_val int;
BEGIN
  DROP TABLE IF EXISTS tmp_tbl;
  CREATE TEMP TABLE tmp_tbl(id int);
  INSERT INTO tmp_tbl values (f1),(10001),(10002);
  SELECT INTO min_val MIN(id) FROM tmp_tbl;
  RAISE INFO 'min_val = %, f2 = %', min_val, f2;
END;
$$ LANGUAGE plpgsql;
```

> **Note**
>
> Lorsque vous écrivez des procédures stockées, nous vous recommandons une bonne pratique pour sécuriser les valeurs sensibles :
> Ne codez pas en dur des informations sensibles dans la logique des procédures stockées. Par exemple, n'attribuez pas de mot de passe utilisateur dans une instruction CREATE USER dans le corps d'une procédure stockée. Cela présente un risque de sécurité, car les valeurs codées en dur peuvent être enregistrées sous forme de métadonnées de schéma dans les

tables du catalogue. Transmettez plutôt des valeurs sensibles, telles que des mots de passe, en tant qu'arguments à la procédure stockée, au moyen de paramètres.
Pour plus d'informations sur les procédures stockées, consultez PROCÉDURE DE CRÉATION et Création de procédures stockées dans Amazon Redshift. Pour plus d'informations sur les tables catalogue, consultez Tables de catalogue système.

L'exemple suivant crée une procédure avec un paramètre IN, un paramètre OUT et un paramètre INOUT.

```
CREATE OR REPLACE PROCEDURE test_sp2(f1 IN int, f2 INOUT varchar(256), out_var OUT
 varchar(256))
AS $$
DECLARE
  loop_var int;
BEGIN
  IF f1 is null OR f2 is null THEN
    RAISE EXCEPTION 'input cannot be null';
  END IF;
  DROP TABLE if exists my_etl;
  CREATE TEMP TABLE my_etl(a int, b varchar);
    FOR loop_var IN 1..f1 LOOP
        insert into my_etl values (loop_var, f2);
        f2 := f2 || '+' || f2;
    END LOOP;
  SELECT INTO out_var count(*) from my_etl;
END;
$$ LANGUAGE plpgsql;
```

L'exemple suivant crée une procédure qui utilise le SECURITY DEFINER paramètre. Cette procédure s'exécute en utilisant les privilèges de l'utilisateur propriétaire de la procédure.

```
CREATE OR REPLACE PROCEDURE sp_get_current_user_definer()
AS $$
DECLARE curr_user varchar(250);
BEGIN
  SELECT current_user INTO curr_user;
  RAISE INFO '%', curr_user;
END;
$$ LANGUAGE plpgsql
SECURITY DEFINER;
```

L'exemple suivant crée une procédure qui utilise le SECURITY INVOKER paramètre. Cette procédure s'exécute en utilisant les privilèges de l'utilisateur qui exécute la procédure.

```
CREATE OR REPLACE PROCEDURE sp_get_current_user_invoker()
AS $$
DECLARE curr_user varchar(250);
BEGIN
  SELECT current_user INTO curr_user;
  RAISE INFO '%', curr_user;
END;
$$ LANGUAGE plpgsql
SECURITY INVOKER;
```

CREATE RLS POLICY

Crée une nouvelle politique de sécurité au niveau des lignes pour fournir un accès précis aux objets de base de données.

Les super-utilisateurs, les utilisateurs ou les utilisateurs disposant du rôle sys:secadmin peuvent créer une stratégie.

Syntaxe

```
CREATE RLS POLICY policy_name
[ WITH (column_name data_type [, ...]) [ [AS] relation_alias ] ]
USING ( using_predicate_exp )
```

Paramètres

policy_name

Nom de la politique .

WITH (column_name data_type [, ...])

Spécifie column_name et data_type référencés aux colonnes des tables auxquelles la politique est attachée.

Vous pouvez omettre la clause WITH uniquement lorsque la politique RLS ne fait référence à aucune colonne des tables à laquelle la politique est attachée.

AS relation_alias

Spécifie un alias facultatif pour la table à laquelle la politique RLS sera attachée.

USING (using_predicate_exp)

Spécifie un filtre qui est appliqué à la clause WHERE de la requête. Amazon Redshift applique un prédicat de politique avant les prédicats utilisateur au niveau de la requête. Par exemple, **`current_user = 'joe' and price > 10`** permet à Joe de ne voir que les enregistrements dont le prix est supérieur à 10 $.

Notes d'utilisation

Lorsque vous utilisez l'instruction CREATE RLS POLICY, tenez compte des points suivants :

- Amazon Redshift prend en charge les filtres qui peuvent faire partie de la clause WHERE d'une requête.
- Toutes les politiques attachées à une table doivent avoir été créées avec le même alias de table.
- Vous n'avez pas besoin de l'autorisation SELECT sur les tables de recherche. Lorsque vous créez une stratégie, Amazon Redshift accorde l'autorisation SELECT à la table de recherche pour la politique correspondante. Une table de recherche est un objet de table utilisé dans une définition de stratégie.
- La sécurité au niveau des lignes d'Amazon Redshift ne prend pas en charge les types d'objets suivant dans une définition de politique : les tables de catalogue, les relations entre bases de données, les tables externes, les vues régulières, les vues à liaison tardive, les tables avec des politiques RLS activées et les tables temporaires.

Exemples

Les instructions SQL suivantes créent les tables, les utilisateurs et les rôles pour l'exemple CREATE RLS POLICY.

```
-- Create users and roles reference in the policy statements.
CREATE ROLE analyst;

CREATE ROLE consumer;
```

```
CREATE USER bob WITH PASSWORD 'Name_is_bob_1';

CREATE USER alice WITH PASSWORD 'Name_is_alice_1';

CREATE USER joe WITH PASSWORD 'Name_is_joe_1';

GRANT ROLE sys:secadmin TO bob;

GRANT ROLE analyst TO alice;

GRANT ROLE consumer TO joe;

GRANT ALL ON TABLE tickit_category_redshift TO PUBLIC;
```

L'exemple suivant crée une politique appelée policy_concerts.

```
CREATE RLS POLICY policy_concerts
WITH (catgroup VARCHAR(10))
USING (catgroup = 'Concerts');
```

CREATE ROLE

Crée un nouveau rôle personnalisé qui est un ensemble d'autorisations. Pour obtenir une liste des rôles définis par le système Amazon Redshift, consultez the section called "Rôles définis par le système Amazon Redshift". Interrogez SVV_ROLES pour afficher les rôles actuellement créés dans votre cluster ou groupe de travail.

Il existe un quota quant au nombre de rôles pouvant être créés. Pour plus d'informations, consultez Quotas et limites dans Amazon Redshift dans le Guide de gestion Amazon Redshift.

Autorisations requises

Les privilèges suivants sont requis pour CREATE ROLE.

- Superuser
- Utilisateurs disposant du privilège CREATE ROLE

Syntaxe

```
CREATE ROLE role_name
```

```
[ EXTERNALID external_id ]
```

Paramètres

role_name

Nom du rôle. Le nom du rôle doit être unique et ne peut être identique à aucun nom d'utilisateur. Un nom de rôle ne peut pas être un mot réservé.

Un super-utilisateur ou un utilisateur standard disposant du privilège CREATE ROLE peut créer des rôles. Un utilisateur qui n'est pas un super-utilisateur, mais qui s'est vu accorder les privilèges USAGE pour WITH GRANT OPTION et ALTER peut accorder ce rôle à n'importe qui.

EXTERNALID external_id

Identificateur du rôle associé à un fournisseur d'identité. Pour plus d'informations, consultez Fédération de fournisseur d'identité natif pour Amazon Redshift.

Exemples

L'exemple suivant crée un rôle `sample_role1`.

```
CREATE ROLE sample_role1;
```

L'exemple suivant crée un rôle `sample_role1`, avec un ID externe associé à un fournisseur d'identité.

```
CREATE ROLE sample_role1 EXTERNALID "ABC123";
```

CREATE SCHEMA

Définit un nouveau schéma pour la base de données actuelle.

Privilèges requis

Les privilèges suivants sont requis pour CREATE SCHEMA :

- Superuser
- Utilisateurs disposant du privilège CREATE SCHEMA

Syntaxe

```
CREATE SCHEMA [ IF NOT EXISTS ] schema_name [ AUTHORIZATION username ]
           [ QUOTA {quota [MB | GB | TB] | UNLIMITED} ] [ schema_element [ ... ]

CREATE SCHEMA AUTHORIZATION username[ QUOTA {quota [MB | GB | TB] | UNLIMITED} ]
 [ schema_element [ ... ] ]
```

Paramètres

IF NOT EXISTS

Clause indiquant que si le schéma spécifié existe déjà, la commande ne doit faire aucune modification et renvoyer un message selon lequel le schéma existe, plutôt que de mettre fin avec une erreur.

Comme cette clause est utile lors de l'écriture de scripts, le script n'échoue pas si CREATE SCHEMA tente de créer un schéma qui existe déjà.

nom_schéma

Nom du nouveau schéma. Le nom du schéma ne peut pas être PUBLIC. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

> **Note**
>
> La liste des schémas du paramètre de configuration search_path détermine la priorité des objets portant le même nom quand il y est fait référence sans noms de schéma.

AUTHORIZATION

Clause qui accorde la propriété à un utilisateur spécifié.

nom d'utilisateur

Nom du propriétaire du schéma.

élément_schéma

Définition d'un ou de plusieurs objets à créer dans le schéma.

QUOTA

Quantité maximale d'espace disque que le schéma spécifié peut utiliser. Cet espace est l'utilisation collective du disque. Il inclut toutes les tables permanentes, les vues matérialisées sous le schéma spécifié et les copies dupliquées de toutes les tables avec la distribution ALL sur chaque nœud de calcul. Le quota de schéma ne prend pas en compte les tables temporaires créées dans le cadre d'un espace de noms ou d'un schéma temporaire.

Pour afficher les quotas de schéma configurés, consultez SVV_SCHEMA_QUOTA_STATE.

Pour afficher les enregistrements où les quotas de schéma ont été dépassés, consultez STL_SCHEMA_QUOTA_VIOLATIONS.

Amazon Redshift convertit la valeur sélectionnée en mégaoctets. Le gigaoctet est l'unité de mesure par défaut lorsque vous ne spécifiez pas de valeur.

Vous devez être un superutilisateur de base de données pour définir et modifier un quota de schéma. Un utilisateur qui n'est pas un superutilisateur mais qui dispose de l'autorisation CREATE SCHEMA peut créer un schéma avec un quota défini. Lorsque vous créez un schéma sans définir de quota, le schéma dispose d'un quota illimité. Lorsque vous définissez le quota en dessous de la valeur actuelle utilisée par le schéma, Amazon Redshift ne permet pas d'ingestion supplémentaire tant que vous n'avez pas libéré d'espace disque. Une instruction DELETE supprime les données d'une table et l'espace disque n'est libéré que lorsque VACUUM s'exécute.

Amazon Redshift vérifie la présence de violations de quotas dans chaque transaction avant de valider la transaction. Amazon Redshift compare la taille (l'espace disque utilisé par toutes les tables d'un schéma) de chaque schéma modifié au quota défini. Étant donné que la vérification des violations de quota se produit à la fin d'une transaction, la limite de taille peut dépasser temporairement le quota dans une transaction avant qu'il ne soit validé. Lorsqu'une transaction dépasse le quota, Amazon Redshift abandonne la transaction, interdit les ingestions ultérieures et rétablit toutes les modifications jusqu'à ce que vous libériez de l'espace disque. En raison de VACUUM en arrière-plan et du nettoyage interne, il est possible qu'un schéma ne soit pas plein au moment où vous vérifiez le schéma après une transaction annulée.

Dans certains cas exceptionnels, Amazon Redshift ne tient pas compte des violations de quotas et valide les transactions. Amazon Redshift le fait pour les transactions qui consistent uniquement en une ou plusieurs des instructions suivantes lorsqu'il n'y a pas d'instruction d'ingestion INSERT ou COPY dans la même transaction :

- DELETE

- TRUNCATE
- VACUUM
- DROP TABLE
- ALTER TABLE APPEND uniquement lors du déplacement de données du schéma complet vers un autre schéma non complet

UNLIMITED

Amazon Redshift n'impose aucune limite à la croissance de la taille totale du schéma.

Limites

Amazon Redshift applique les limites suivantes pour les schémas.

- Il y a un maximum de 9900 schémas par base de données.

Exemples

L'exemple suivant crée un schéma nommé US_SALES et accorde la propriété à l'utilisateur DWUSER.

```
create schema us_sales authorization dwuser;
```

L'exemple suivant crée un schéma nommé US_SALES, donne la propriété à l'utilisateur DWUSER et définit le quota sur 50 Go.

```
create schema us_sales authorization dwuser QUOTA 50 GB;
```

Pour consulter le nouveau schéma, interrogez la table de catalogue PG_NAMESPACE comme illustré ci-après :

```
select nspname as schema, usename as owner
from pg_namespace, pg_user
where pg_namespace.nspowner = pg_user.usesysid
and pg_user.usename ='dwuser';

   schema |  owner
----------+----------
```

```
 us_sales | dwuser
(1 row)
```

L'exemple suivant crée le schéma US_SALES, ou ne fait rien et renvoie un message si le schéma existe déjà :

```
create schema if not exists us_sales;
```

CREATE TABLE

Crée une nouvelle table dans la base de données actuelle. Vous définissez une liste de colonnes qui contiennent chacune des données d'un type distinct. Le propriétaire de la table est l'émetteur de la commande CREATE TABLE.

Privilèges requis

Les privilèges suivants sont requis pour CREATE TABLE :

- Superuser
- Utilisateurs disposant du privilège CREATE TABLE

Syntaxe

```
CREATE [ [LOCAL ] { TEMPORARY | TEMP } ] TABLE
[ IF NOT EXISTS ] table_name
( { column_name data_type [column_attributes] [ column_constraints ]
  | table_constraints
  | LIKE parent_table [ { INCLUDING | EXCLUDING } DEFAULTS ] }
  [, ... ]  )
[ BACKUP { YES | NO } ]
[table_attributes]

where column_attributes are:
  [ DEFAULT default_expr ]
  [ IDENTITY ( seed, step ) ]
  [ GENERATED BY DEFAULT AS IDENTITY ( seed, step ) ]
  [ ENCODE encoding ]
  [ DISTKEY ]
  [ SORTKEY ]
  [ COLLATE CASE_SENSITIVE | COLLATE CASE_INSENSITIVE  ]
```

```
and column_constraints are:
  [ { NOT NULL | NULL } ]
  [ { UNIQUE  |  PRIMARY KEY } ]
  [ REFERENCES reftable [ ( refcolumn ) ] ]

and table_constraints  are:
  [ UNIQUE ( column_name [, ... ] ) ]
  [ PRIMARY KEY ( column_name [, ... ] )  ]
  [ FOREIGN KEY (column_name [, ... ] ) REFERENCES reftable [ ( refcolumn ) ]


and table_attributes are:
  [ DISTSTYLE { AUTO | EVEN | KEY | ALL } ]
  [ DISTKEY ( column_name ) ]
  [ [COMPOUND | INTERLEAVED ] SORTKEY ( column_name [,...]) |  [ SORTKEY AUTO ] ]
  [ ENCODE AUTO ]
```

Paramètres

LOCAL

Facultatif. Même si ce mot-clé est accepté dans l'instruction, il n'a aucun effet dans Amazon Redshift.

TEMPORARY | TEMP

Mot-clé qui crée une table temporaire visible uniquement dans la séance en cours. La table temporaire est automatiquement supprimée à la fin de la séance dans laquelle elle a été créée. La table temporaire peut avoir le même nom qu'une table permanente. La table temporaire est créée dans un schéma distinct, propre à la séance. (Vous ne pouvez pas spécifier un nom pour ce schéma.) Comme ce schéma temporaire devient le premier schéma du chemin de recherche, la table temporaire a priorité sur la table permanente, sauf si vous qualifiez le nom de la table avec le nom du schéma pour accéder à la table permanente. Pour plus d'informations sur les schémas et les priorités, consultez search_path.

> **Note**
>
> Par défaut, les utilisateurs de la base de données ont l'autorisation de créer des tables temporaires par leur appartenance automatique au groupe PUBLIC. Pour refuser ce privilège à un utilisateur, retirez le privilège TEMP du groupe PUBLIC, puis accordez

explicitement le privilège TEMP uniquement à des utilisateurs ou groupes d'utilisateurs spécifiques.

IF NOT EXISTS

Clause indiquant que si la table spécifiée existe déjà, la commande ne doit faire aucune modification et renvoyer un message selon lequel la table existe, plutôt que de s'arrêter avec une erreur. Notez que la table existante peut ne ressembler en rien à celle que vous avez créée ; seul le nom de la table est comparé.

Comme cette clause est utile lors de l'écriture de scripts, le script n'échoue pas si CREATE TABLE tente de créer une table qui existe déjà.

table_name

Nom de la table à créer.

Important

Si vous spécifiez un nom de table qui commence par « # », la table est créée comme table temporaire. Voici un exemple :

```
create table #newtable (id int);
```

Vous faites également référence à la table avec le « # ». Par exemple :

```
select * from #newtable;
```

La longueur maximale d'un nom de table est de 127 octets ; les noms plus longs sont tronqués à 127 octets. Vous pouvez utiliser des caractères multioctets UTF-8 jusqu'à un maximum de quatre octets. Amazon Redshift applique un quota du nombre de tables par cluster et par type de nœud, y compris les tables temporaires définies par l'utilisateur et les tables temporaires créées par Amazon Redshift lors du traitement des requêtes ou de la maintenance du système. Le cas échéant, le nom de la table peut être qualifié avec le nom de la base de données et le nom du schéma. Dans l'exemple suivant, le nom de base de données est `tickit`, le nom du schéma `public` et le nom de la table `test`.

```
create table tickit.public.test (c1 int);
```

Si la base de données ou le schéma n'existe pas, la table n'est pas créée et l'instruction renvoie une erreur. Vous ne pouvez pas créer de tables ni de vues dans les bases de données système `template0`, `template1`, `padb_harvest` ou `sys:internal`.

Si un nom de schéma est donné, la nouvelle table est créée dans ce schéma (en supposant que le créateur ait accès au schéma). Le nom de la table doit être un nom unique pour ce schéma. Si aucun schéma n'est spécifié, la table est créée en utilisant le schéma de base de données actuel. Si vous créez une table temporaire, vous ne pouvez pas spécifier un nom de schéma, car les tables temporaires existent dans un schéma spécial.

Plusieurs tables temporaires de même nom peuvent exister en même temps dans la même base de données même si elles sont créées dans des séances distinctes, car les tables sont attribuées à des schémas différents. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

column_name

Nom d'une colonne à créer dans la nouvelle table. La longueur maximale d'un nom de colonne est de 127 octets ; les noms plus longs sont tronqués à 127 octets. Vous pouvez utiliser des caractères multioctets UTF-8 jusqu'à un maximum de quatre octets. Le nombre maximal de colonnes que vous pouvez définir dans une seule table est de 1 600. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

> **Note**
>
> Si vous créez une grande table, veillez à ce que la liste de colonnes ne dépasse pas les limites de largeur de ligne pour les résultats intermédiaires pendant le traitement des charges et des requêtes. Pour plus d'informations, consultez Notes d'utilisation.

data_type

Type de données de la colonne en cours de création. Pour les colonnes CHAR et VARCHAR, vous pouvez utiliser le mot-clé MAX au lieu de déclarer une longueur maximale. MAX définit la longueur maximale sur 4 096 octets pour CHAR ou 65535 octets pour VARCHAR. La taille maximum d'un objet GEOMETRY est de 1 048 447 octets.

Pour obtenir des informations sur les types de données pris en charge par Amazon Redshift, consultez Types de données.

DEFAULT expr_défaut

Clause qui attribue une valeur de données par défaut à la colonne. Le type de données de expr_défaut doit correspondre au type de données de la colonne. La valeur DEFAULT doit être une expression exempte de variable. Les sous-requêtes, les références croisées aux autres colonnes de la table active et les fonctions définies par l'utilisateur ne sont pas autorisées.

L'expression expr_défaut est utilisée dans toute opération INSERT qui ne spécifie pas une valeur pour la colonne. Si aucune valeur par défaut n'est spécifiée, la valeur par défaut de la colonne est la valeur null.

Si une opération COPY avec une liste de colonnes définies omet une colonne qui a une valeur DEFAULT, la commande COPY insère la valeur de l'expression expr_défaut.

IDENTITY(seed, step)

Clause qui spécifie que la colonne est une colonne IDENTITY. Une colonne IDENTITY contient des valeurs uniques générées automatiquement. Le type de données d'une colonne IDENTITY doit être INT ou BIGINT.

Lorsque vous ajoutez des lignes à l'aide d'une instruction `INSERT` ou `INSERT INTO [tablename] VALUES()`, ces valeurs commencent par la valeur spécifiée en tant que seed et sont incrémentées du nombre spécifié comme step.

Quand la table est chargée à l'aide d'une instruction `INSERT INTO [tablename] SELECT * FROM` ou `COPY`, les données sont chargées en parallèle et distribuées aux tranches de nœuds. Pour garantir que les valeurs d'identité sont uniques, Amazon Redshift ignore un certain nombre de valeurs lors de la création des valeurs d'identité. Les valeurs d'identité sont uniques, mais l'ordre ne correspond pas toujours à celui des fichiers source.

GENERATED BY DEFAULT AS IDENTITY(seed, step)

Clause qui spécifie que la colonne est une colonne IDENTITY par défaut et qui vous permet d'attribuer automatiquement une valeur unique à la colonne. Le type de données d'une colonne IDENTITY doit être INT ou BIGINT. Lorsque vous ajoutez des lignes sans valeurs, ces valeurs commencent par la valeur spécifiée en tant que seed et s'incrémentent du nombre spécifié comme s step. Pour obtenir des informations sur la manière dont les valeurs sont générées, consultez IDENTITY.

De plus, lors des opérations INSERT, UPDATE ou COPY, vous pouvez fournir une valeur sans EXPLICIT_IDS. Amazon Redshift utilise cette valeur pour insérer dans la colonne d'identité au lieu d'utiliser la valeur générée par le système. La valeur peut être un doublon, une valeur inférieure au seed ou une valeur comprise entre les valeurs de step. Amazon Redshift ne vérifie pas si les valeurs de la colonne sont uniques. Le fait de fournir une valeur n'affecte pas la prochaine valeur générée par le système.

> **Note**
>
> Si vous avez besoin de valeurs uniques dans la colonne, n'ajoutez pas de valeur en double. À la place, ajoutez une valeur unique inférieure au seed ou comprise entre les valeurs de step.

Gardez à l'esprit ce qui suit au sujet des colonnes d'identité par défaut :

- Les colonnes d'identité par défaut sont NOT NULL. NULL ne peut pas être inséré.
- Pour insérer une valeur générée dans une colonne d'identité par défaut, utilisez le mot-clé DEFAULT.

```
INSERT INTO tablename (identity-column-name) VALUES (DEFAULT);
```

- Le fait de remplacer des valeurs d'une colonne d'identité par défaut n'affecte pas la prochaine valeur générée.
- Vous ne pouvez pas ajouter de colonne d'identité par défaut avec l'instruction ALTER TABLE ADD COLUMN.
- Vous pouvez ajouter une colonne d'identité par défaut avec l'instruction ALTER TABLE APPEND.

ENCODE encodage

Encodage de compression pour une colonne. ENCODE AUTO est la valeur par défaut pour les tables. Amazon Redshift gère automatiquement l'encodage de compression pour toutes les colonnes de la table. Si vous spécifiez l'encodage de compression pour une colonne quelconque du tableau, le tableau n'est plus défini sur ENCODE AUTO. Amazon Redshift ne gère plus automatiquement le codage de compression pour toutes les colonnes de la table. Vous pouvez spécifier l'option ENCODE AUTO pour la table afin de permettre à Amazon Redshift de gérer automatiquement l'encodage de la compression pour toutes les colonnes de la table.

Pour les colonnes pour lesquelles vous ne spécifiez pas d'encodage de compression, Amazon Redshift affecte automatiquement un encodage de compression initial comme suit :

- Toutes les colonnes de tables temporaires se voient attribuer une compression RAW par défaut.
- Les colonnes qui sont définies comme des clés de tri se voient attribuer une compression RAW.
- Les colonnes qui sont définies comme des types de données BOOLEAN, REAL, DOUBLE PRECISION ou GEOMETRY ou GEOGRAPHY se voient attribuer une compression RAW.
- Les colonnes définies comme SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ sont soumises à une compression. AZ64
- Les colonnes définies comme CHAR, VARCHAR ou VARBYTE sont affectées à la compression LZO.

> **Note**
>
> Si vous ne souhaitez pas qu'une colonne soit compressée, spécifiez explicitement un encodage RAW (brut).

Les encodages compression encodings (p. 69) suivants sont pris en charge :

- AZ64
- BYTEDICT
- DELTA
- DELTA32 K
- LZO
- MOSTLY8
- MOSTLY16
- MOSTLY32
- RAW (aucune compression)
- RUNLENGTH
- TEXT255
- TEXT32 K

- ZSTD

DISTKEY

Mot-clé qui spécifie que la colonne est la clé de distribution de la table. Une seule colonne d'une table peut être la clé de distribution. Vous pouvez utiliser le mot-clé DISTKEY après un nom de colonne ou dans le cadre de la définition de la table à l'aide de la syntaxe DISTKEY (nom_colonne). Les deux méthodes ont le même effet. Pour plus d'informations, consultez le paramètre DISTSTYLE ultérieurement dans cette rubrique.

Le type de données d'une colonne de clés de distribution peut être : REAL, DOUBLE PRECISION, SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP, TIMESTAMPTZ, CHAR ou VARCHAR.

SORTKEY

Mot-clé qui spécifie que la colonne est la clé de tri de la table. Lorsque les données sont chargées dans la table, les données sont triées sur une ou plusieurs colonnes désignées comme les clés de tri. Vous pouvez utiliser le mot-clé SORTKEY après un nom de colonne pour spécifier une clé de tri à une seule colonne ou vous pouvez spécifier une ou plusieurs colonnes comme colonnes de clé de tri de la table à l'aide de la syntaxe SORTKEY (column_name [,...]). Seules les clés de tri composées sont créées avec cette syntaxe.

Vous pouvez définir un maximum de 400 colonnes SORTKEY par table.

Le type de données d'une colonne de clés de tri peut être : BOOLEAN, REAL, DOUBLE PRECISION, SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP, TIMESTAMPTZ, CHAR ou VARCHAR.

COLLATE CASE_SENSITIVE | COLLATE CASE_INSENSITIVE

Clause qui spécifie si la recherche de chaîne ou la comparaison sur la colonne est CASE_SENSITIVE (sensible à la casse) ou CASE_INSENSITIVE (insensible à la casse). La valeur par défaut est la même que la configuration actuelle de sensibilité à la casse de la base de données.

Pour rechercher les informations de classement de la base de données, utilisez la commande suivante :

```
SELECT db_collation();
```

```
db_collation
----------------
 case_sensitive
(1 row)
```

NOT NULL | NULL

NOT NULL spécifie que la colonne n'est pas autorisée à contenir des valeurs null. NULL, valeur par défaut, spécifie que la colonne accepte les valeurs null. Les colonnes IDENTITY sont déclarées NOT NULL par défaut.

UNIQUE

Mot-clé qui spécifie que la colonne ne peut contenir que des valeurs uniques. Le comportement de la contrainte de table unique est identique à celui des contraintes de colonne, avec la capacité supplémentaire de s'étendre sur plusieurs colonnes. Pour définir une contrainte de table unique, utilisez la syntaxe UNIQUE (nom_colonne [,...]).

> **Important**
>
> Les contraintes d'unicité ont une valeur informationnelle et ne sont pas appliquées par le système.

PRIMARY KEY

Mot-clé qui spécifie que la colonne est la clé primaire de la table. Seule une colonne peut être définie comme clé primaire à l'aide d'une définition de colonne. Pour définir une contrainte de table avec une clé primaire à plusieurs colonnes, utilisez la syntaxe PRIMARY KEY (nom_colonne [,...]).

L'identification d'une colonne comme clé primaire fournit les métadonnées relatives à la conception du schéma. Une clé primaire implique que d'autres tables puissent se reposer sur cet ensemble de colonnes comme identificateur unique des lignes. Une clé primaire peut être spécifiée pour une table, que ce soit comme contrainte de colonne ou contrainte de table. La contrainte de clé primaire doit nommer un ensemble de colonnes différent des autres ensembles de colonnes nommés pour une contrainte unique définie pour la même table.

Les colonnes PRIMARY KEY sont également définies comme NOT NULL.

> **Important**
>
> Les contraintes de clé primaire ont uniquement un but informatif. Elles ne sont pas appliquées par le système, mais sont utilisées par le planificateur.

References table_réf [(colonne_réf)]

Clause qui spécifie une contrainte de clé étrangère, ce qui implique que la colonne contienne uniquement des valeurs qui correspondent à celles de la colonne référencée d'une ligne de la table référencée. Les colonnes référencées doivent être les colonnes d'une contrainte de clé unique ou de clé primaire de la table référencée.

> **Important**
>
> Les contraintes de clé étrangère ont un but informatif uniquement. Elles ne sont pas appliquées par le système, mais sont utilisées par le planificateur.

LIKE table_parent [{ INCLUDING | EXCLUDING } DEFAULTS]

Clause qui spécifie une table existante à partir de laquelle la nouvelle table copie automatiquement les noms de colonne, les types de données et les contraintes NOT NULL. La nouvelle table et la table parent sont découplées et toutes les modifications apportées à la table parent ne sont pas appliquées à la nouvelle table. Les expressions par défaut des définitions de colonne copiées sont copiées uniquement si INCLUDING DEFAULTS est spécifié. Le comportement par défaut consiste à exclure les expressions par défaut, de telle sorte que toutes les colonnes de la nouvelle table aient une valeurs null par défaut.

Les tables créées avec l'option LIKE n'héritent pas des contraintes de clé primaire et de clé étrangère. Les propriétés du style de distribution, des clés de tri, de BACKUP et de NULL sont héritées par les tables LIKE, mais vous ne pouvez pas les définir explicitement dans l'instruction CREATE TABLE ... LIKE.

BACKUP { YES | NO }

Clause qui spécifie si la table doit être incluse dans les instantanés de cluster automatiques et manuels.

Pour les tables, telles que les tables intermédiaires, qui ne contiennent pas de données critiques, spécifiez BACKUP NO pour économiser du temps de traitement lorsque la création d'instantanés et la restauration à partir d'instantanés, et pour réduire l'espace de stockage sur Amazon Simple Storage Service. Le paramètre BACKUP NO n'a aucun effet sur la réplication automatique des données vers les autres nœuds du cluster. Les tables pour lesquelles BACKUP NO est spécifié sont donc restaurées en cas de défaillance d'un nœud. La valeur par défaut est BACKUP YES.

> **Note**
>
> Les tables sans sauvegarde ne sont pas prises en charge pour RA3 et sans serveur. Une table marquée comme non sauvegardée RA3 et Serverless sera traitée comme une table permanente qui sera toujours sauvegardée lors de la prise d'un instantané et restaurée lors de la restauration à partir d'un instantané.

DISTSTYLE { AUTO | EVEN | KEY | ALL }

Mot-clé qui définit le style de distribution des données pour l'ensemble de la table. Amazon Redshift répartit les lignes d'une table entre les nœuds de calcul selon le style de distribution spécifié pour la table. La valeur par défaut est AUTO.

Le style de distribution que vous sélectionnez pour les tables affecte les performances globales de votre base de données. Pour plus d'informations, consultez Distribution des données pour l'optimisation des requêtes. Les styles de distribution possibles sont les suivants :

- AUTO : Amazon Redshift attribue un style de distribution optimal basé sur les données de table. Par exemple, si le style de distribution AUTO est spécifié, Amazon Redshift affecte initialement le style de distribution ALL à une petite table. Lorsque la table s'agrandit, Amazon Redshift peut modifier le style de distribution sur KEY et choisir la clé primaire (ou une colonne de la clé primaire composite) comme DISTKEY. Si la table s'agrandit et qu'aucune des colonnes ne peut être DISTKEY, Amazon Redshift change le style de distribution sur EVEN. La modification de style de distribution se produit en arrière-plan et son impact sur les requêtes des utilisateurs est minimal.

  Pour consulter le style de distribution appliqué à une table, interrogez la table catalogue système PG_CLASS. Pour plus d'informations, consultez Affichage des styles de distribution.
- EVEN : les données de la table sont réparties également entre les nœuds d'un cluster dans une distribution en tourniquet (round robin). IDs Les lignes sont utilisées pour déterminer la distribution, et approximativement le même nombre de lignes est distribué à chaque nœud.

- KEY : les données sont réparties selon les valeurs de la colonne DISTKEY. Lorsque vous définissez les colonnes de jointure des tables de jointure comme clés de distribution, les lignes de jointure des deux tables sont colocalisées sur les nœuds de calcul. Lorsque les données sont colocalisées, l'optimiseur peut effectuer les jointures plus efficacement. Si vous spécifiez DISTSTYLE KEY, vous devez nommer une colonne DISTKEY, pour la table ou dans le cadre de la définition de colonne. Pour plus d'informations, consultez le paramètre DISTKEY plus haut dans cette rubrique.
- ALL : une copie de la table complète est distribuée sur chaque nœud. Ce style de distribution garantit que toutes les lignes obligatoires pour une jointure sont disponibles sur chaque nœud, mais il multiplie les besoins de stockage et augmente les temps de charge et de maintenance de la table. Une distribution ALL peut améliorer le temps d'exécution lorsqu'elle est utilisée avec certaines tables de dimension où la distribution KEY ne convient pas, mais les améliorations des performances doivent être pondérées par rapport aux coûts de maintenance.

DISTKEY (nom_colonne)

Contrainte qui spécifie la colonne à utiliser comme clé de distribution de la table. Vous pouvez utiliser le mot-clé DISTKEY après un nom de colonne ou dans le cadre de la définition de la table à l'aide de la syntaxe DISTKEY (nom_colonne). Les deux méthodes ont le même effet. Pour plus d'informations, consultez le paramètre DISTSTYLE plus haut dans cette rubrique.

[COMPOUND | INTERLEAVED] SORTKEY (column_name [,...]) | [SORTKEY AUTO]

Spécifie une ou plusieurs clés de tri pour la table. Lorsque les données sont chargées dans la table, les données sont triées sur les colonnes désignées comme clés de tri. Vous pouvez utiliser le mot-clé SORTKEY après un nom de colonne pour spécifier une clé de tri à une seule colonne ou vous pouvez spécifier une ou plusieurs colonnes comme colonnes de clé de tri de la table à l'aide de la syntaxe `SORTKEY (column_name [ , ... ] )`.

Vous pouvez spécifier le cas échéant le style de tri COMPOUND ou INTERLEAVED. Si vous spécifiez SORTKEY avec des colonnes, la valeur par défaut est COMPUND. Pour plus d'informations, consultez Clés de tri.

Si vous ne spécifiez pas d'options de clé de tri, la valeur par défaut est AUTO.

Vous pouvez définir un maximum de 400 colonnes COMPOUND SORTKEY ou de 8 colonnes INTERLEAVED SORTKEY par table.

AUTO

Spécifie qu'Amazon Redshift attribue une clé de tri optimale en fonction des données de la table. Par exemple, si la clé de tri AUTO est spécifiée, Amazon Redshift n'affecte initialement aucune clé de tri à une table. Si Amazon Redshift détermine qu'une clé de tri améliorera les performances des requêtes, Amazon Redshift peut modifier la clé de tri de votre table. Le tri de la table est effectué par tri automatique de la table. Pour plus d'informations, consultez Tri automatique des tables.

Amazon Redshift ne modifie pas les tables qui possèdent des clés de tri ou de distribution existantes, à une exception près : si une table possède une clé de distribution qui n'a jamais été utilisée dans une requête JOIN, la clé peut être modifiée si Amazon Redshift détermine qu'il en existe une meilleure.

Pour afficher la clé de tri d'une table, interrogez la vue catalogue système SVV__TABLE_INFO. Pour plus d'informations, consultez SVV_TABLE_INFO. Pour afficher les recommandations Amazon Redshift Advisor pour les tables, recherchez la vue catalogue système SVV_ALTER_TABLE_RECOMMENDATIONS. Pour plus d'informations, consultez SVV_ALTER_TABLE_RECOMMENDATIONS. Pour afficher les actions effectuées par Amazon Redshift, interrogez la vue catalogue système SVL_AUTO_WORKER_ACTION. Pour plus d'informations, consultez SVL_AUTO_WORKER_ACTION.

COMPOUND

Spécifie que les données sont triées à l'aide d'une clé composée, constituée de toutes les colonnes affichées, dans leur ordre d'apparition. Une clé de tri composée est surtout utile lorsqu'une requête analyse les lignes selon l'ordre des colonnes de tri. Les avantages en termes de performances d'un tri avec une clé composée diminuent lorsque les requêtes reposent sur des colonnes de tri secondaires. Vous pouvez définir un maximum de 400 colonnes COMPOUND SORTKEY par table.

INTERLEAVED

Spécifie que les données sont triées à l'aide d'une clé de tri entrelacée. Un maximum de huit colonnes peut être spécifié pour une clé de tri entrelacée.

Comme un tri entrelacé confère un poids égal à chaque colonne, ou sous-ensemble de colonnes, de la clé de tri, les requêtes ne dépendent pas de l'ordre des colonnes de la clé de tri. Quand une requête utilise une ou plusieurs colonnes de tri secondaires, le tri entrelacé améliore les performances des requêtes de façon significative. Le tri entrelacé entraîne un

léger coût de traitement pour les opérations de chargement de données et les opérations VACUUM.

> **Important**
>
> N'utilisez pas de clé de tri entrelacée sur les colonnes qui contiennent des attributs qui augmentent de façon monotone, tels que les colonnes d'identité, les dates ou les horodatages.

ENCODE AUTO

Permet à Amazon Redshift d'ajuster automatiquement le type d'encodage pour toutes les colonnes de la table afin d'optimiser les performances des requêtes. ENCODE AUTO conserve les types d'encodage initiaux que vous spécifiez lors de la création de la table. Ensuite, si Amazon Redshift détermine qu'un nouveau type d'encodage peut améliorer les performances de la requête, Amazon Redshift peut modifier le type d'encodage des colonnes de la table. ENCODE AUTO est la valeur par défaut si vous ne spécifiez de type d'encodage pour aucune colonne de la table.

UNIQUE (nom_colonne [,...])

Contrainte qui spécifie qu'un groupe d'une ou de plusieurs colonnes d'une table ne peut contenir que des valeurs uniques. Le comportement de la contrainte de table unique est identique à celui des contraintes de colonne, avec la capacité supplémentaire de s'étendre sur plusieurs colonnes. Dans le contexte des contraintes uniques, les valeurs null ne sont pas considérées comme égales. Chaque contrainte de table unique doit nommer un ensemble de colonnes différent de l'ensemble de colonnes nommées par une autre contrainte de clé unique ou de clé primaire définie pour la table.

> **Important**
>
> Les contraintes d'unicité ont une valeur informationnelle et ne sont pas appliquées par le système.

PRIMARY KEY (nom_colonne [,...])

Contrainte qui spécifie qu'une colonne ou un nombre de colonnes d'une table ne peut contenir que des valeurs non null (non dupliquées) uniques. L'identification d'un ensemble de colonnes

comme clé primaire fournit aussi les métadonnées relatives à la conception du schéma. Une clé primaire implique que d'autres tables puissent se reposer sur cet ensemble de colonnes comme identificateur unique des lignes. Une clé primaire peut être spécifiée pour une table, que ce soit comme contrainte de colonne ou contrainte de table. La contrainte de clé primaire doit nommer un ensemble de colonnes différent des autres ensembles de colonnes nommés pour une contrainte unique définie pour la même table.

> **Important**
>
> Les contraintes de clé primaire ont uniquement un but informatif. Elles ne sont pas appliquées par le système, mais sont utilisées par le planificateur.

FOREIGN KEY (nom_colonne [, ...]) REFERENCES table_réf [(colonne_réf)]

Contrainte qui spécifie une contrainte de clé étrangère, laquelle nécessite qu'un groupe d'une ou de plusieurs colonnes de la nouvelle table ne doit contenir que des valeurs qui correspondent à des valeurs des colonnes référencées d'une ligne de la table référencée. Si colonne_réf est omis, la clé primaire de table_réf est utilisée. Les colonnes référencées doivent être les colonnes d'une contrainte de clé unique ou de clé primaire de la table référencée.

> **Important**
>
> Les contraintes de clé étrangère ont un but informatif uniquement. Elles ne sont pas appliquées par le système, mais sont utilisées par le planificateur.

Notes d'utilisation

Les contraintes d'unicité, de clé primaire et de clé externe sont uniquement informatives ; elles ne sont pas appliquées par Amazon Redshift lorsque vous remplissez une table. Par exemple, si vous insérez des données dans une table avec des dépendances, l'insertion peut réussir même si elle enfreint la contrainte. Néanmoins, les clés primaires et les clés étrangères servent de conseils de planification et doivent être déclarées si votre processus ETL ou un autre processus de votre application impose leur intégrité. Pour plus d'informations sur la manière de supprimer une table avec des dépendances, consultez DROP TABLE.

Limites et quotas

Tenez compte des limites suivantes lorsque vous créez une table.

- Il existe une limite pour le nombre maximal de tables dans un cluster par type de nœud. Pour plus d'informations, consultez Limites dans le Guide de gestion Amazon Redshift.
- Le nombre maximal de caractères pour un nom de table est 127.
- Le nombre maximal de colonnes que vous pouvez définir dans une seule table est de 1 600.
- Le nombre maximal de colonnes SORTKEY que vous pouvez définir dans une seule table est de 400.

Résumé des paramètres de niveau colonne et des paramètres de niveau table

Plusieurs attributs et paramètres peuvent être définis au niveau colonne ou au niveau table. Dans certains cas, la définition d'un attribut ou d'une contrainte au niveau colonne ou au niveau table a le même effet. Dans d'autres cas, elles produisent des résultats différents.

La liste suivante résume les paramètres de niveau colonne et de niveau de la table :

DISTKEY

Il n'y a pas de différence effective que la définition soit au niveau colonne ou au niveau table.

Si DISTKEY est défini, au niveau colonne ou au niveau table, DISTSTYLE doit être défini sur KEY ou ne pas l'être du tout. DISTSTYLE peut être défini au niveau table uniquement.

SORTKEY

Si la définition est au niveau colonne, SORTKEY doit être une colonne unique. Si SORTKEY est défini au niveau table, une ou plusieurs colonnes peuvent constituer une clé de tri composée ou une clé de tri composite entrelacée.

COLLATE CASE_SENSITIVE | COLLATE CASE_INSENSITIVE

Amazon Redshift ne prend pas en charge la modification de la configuration de sensibilité à la casse pour une colonne. Lorsque vous ajoutez une nouvelle colonne à la table, Amazon Redshift utilise la valeur par défaut pour la sensibilité à la casse. Amazon Redshift ne prend pas en charge le mot clé COLLATE lors de l'ajout d'une nouvelle colonne.

Pour obtenir des informations sur la création de bases de données à l'aide du classement de bases de données, consultez CREATE DATABASE.

Pour obtenir des informations sur la fonction COLLATE, consultez Fonction COLLATE.

UNIQUE

Au niveau colonne, une ou plusieurs clés peuvent être définies comme UNIQUE ; la contrainte UNIQUE s'applique à chaque colonne individuellement. Si UNIQUE est défini au niveau table, une ou plusieurs colonnes peuvent constituer une contrainte UNIQUE composite.

PRIMARY KEY

Si la définition est au niveau colonne, PRIMARY KEY doit être une colonne unique. Si PRIMARY KEY est défini au niveau de la table, une ou plusieurs colonnes peuvent constituent une clé primaire composite.

FOREIGN KEY

Il n'y a pas de différence effective que FOREIGN KEY soit défini au niveau colonne ou au niveau table. Au niveau colonne, la syntaxe est simplement `REFERENCES` table_réf [(colonne_réf)].

Distribution des données entrantes

Lorsque le schéma de distribution de hachage des données entrantes correspond à celui de la table cible, aucune distribution physique des données n'est réellement nécessaire lors du chargement des données. Par exemple, si une clé de distribution est définie pour la nouvelle table et que les données sont insérées à partir d'une autre table qui est distribuée sur la même colonne de clé, les données sont chargées en place, à l'aide des mêmes nœuds et tranches. Cependant, si les tables source et cible sont toutes deux définies sur la distribution EVEN, les données sont redistribuées dans la table cible.

Tables larges

Vous pourrez créer une table très grande, mais ne pas pouvoir effectuer le traitement de requêtes, telles que les instructions INSERT ou SELECT, sur la table. La largeur maximale d'une table avec des colonnes de largeur fixe, telles que CHAR, est de 64 Ko - 1 (soit 65 535 octets). Si une table inclut des colonnes VARCHAR, la table peut avoir une grandeur déclarée plus élevée sans renvoyer une erreur, car la pleine grandeur déclarée des colonnes VARCHARS n'intervient pas dans la limite calculée de traitement des requêtes. La limite effective de traitement des requêtes avec les colonnes VARCHAR varie selon un certain nombre de facteurs.

Si une table est trop large pour l'insertion ou la sélection, vous recevez l'erreur suivante.

```
ERROR:  8001
```

```
DETAIL:  The combined length of columns processed in the SQL statement
exceeded the query-processing limit of 65535 characters (pid:7627)
```

Exemples

Pour des exemples montrant comment utiliser la commande CREATE TABLE, consultez Exemples.

Exemples

Les exemples suivants illustrent différents attributs de colonne et de table des instructions CREATE TABLE Amazon Redshift. Pour plus d'informations sur CREATE TABLE, y compris la définition des paramètres, consultez CREATE TABLE.

De nombreux exemples utilisent des tables et des données provenant de l'échantillon de données TICKIT. Pour plus d'informations, consultez Exemple de base de données.

Vous pouvez faire précéder le nom de la table du nom de la base de données et du nom du schéma dans une commande CREATE TABLE. Par exemple, `dev_database.public.sales`. Le nom de la base de données doit être la base de données à laquelle vous êtes connecté. Toute tentative de création d'objets de base de données dans une autre base de données échoue avec une erreur d'opération non valide.

Créer une table avec une clé de distribution, une clé de tri composée et une compression

L'exemple suivant crée une table SALES dans la base de données TICKIT avec compression définie pour plusieurs colonnes. LISTID est déclaré comme clé de distribution, et LISTID et SELLERID sont déclarés comme clé de tri composée à plusieurs colonnes. Les contraintes de clé primaire et de clé étrangère sont également définies pour la table. Avant de créer la table de l'exemple, vous devrez peut-être ajouter une contrainte UNIQUE à chaque colonne référencée par une clé étrangère, si les contraintes n'existent pas.

```
create table sales(
salesid integer not null,
listid integer not null,
sellerid integer not null,
buyerid integer not null,
eventid integer not null encode mostly16,
dateid smallint not null,
qtysold smallint not null encode mostly8,
pricepaid decimal(8,2) encode delta32k,
```

```
commission decimal(8,2) encode delta32k,
saletime timestamp,
primary key(salesid),
foreign key(listid) references listing(listid),
foreign key(sellerid) references users(userid),
foreign key(buyerid) references users(userid),
foreign key(dateid) references date(dateid))
distkey(listid)
compound sortkey(listid,sellerid);
```

Les résultats sont les suivants :

```
schemaname | tablename | column     | type                          | encoding | distkey
 | sortkey | notnull
-----------+-----------+------------+-------------------------------+----------+---------
+---------+--------
public     | sales     | salesid    | integer                       | lzo      | false
 |       0 | true
public     | sales     | listid     | integer                       | none     | true
 |       1 | true
public     | sales     | sellerid   | integer                       | none     | false
 |       2 | true
public     | sales     | buyerid    | integer                       | lzo      | false
 |       0 | true
public     | sales     | eventid    | integer                       | mostly16 | false
 |       0 | true
public     | sales     | dateid     | smallint                      | lzo      | false
 |       0 | true
public     | sales     | qtysold    | smallint                      | mostly8  | false
 |       0 | true
public     | sales     | pricepaid  | numeric(8,2)                  | delta32k | false
 |       0 | false
public     | sales     | commission | numeric(8,2)                  | delta32k | false
 |       0 | false
public     | sales     | saletime   | timestamp without time zone | lzo      | false
 |       0 | false
```

L’exemple suivant crée une table t1 avec une colonne col1 insensible à la casse.

```
create table T1 (
  col1 Varchar(20) collate case_insensitive
 );
```

```
insert into T1 values ('bob'), ('john'), ('Tom'), ('JOHN'), ('Bob');
```

Interrogez la table.

```
select * from T1 where col1 = 'John';

col1
------
 john
 JOHN
(2 rows)
```

Créer une table à l'aide d'une clé de tri entrelacé

L'exemple suivant crée la table CUSTOMER avec une clé de tri entrelacé.

```
create table customer_interleaved (
  c_custkey     integer        not null,
  c_name        varchar(25)    not null,
  c_address     varchar(25)    not null,
  c_city        varchar(10)    not null,
  c_nation      varchar(15)    not null,
  c_region      varchar(12)    not null,
  c_phone       varchar(15)    not null,
  c_mktsegment      varchar(10)    not null)
diststyle all
interleaved sortkey (c_custkey, c_city, c_mktsegment);
```

Créer une table avec IF NOT EXISTS

L'exemple suivant crée la table CITIES ou ne fait rien et renvoie un message si elle existe déjà :

```
create table if not exists cities(
cityid integer not null,
city varchar(100) not null,
state char(2) not null);
```

Créer une table avec la distribution ALL

L'exemple suivant crée la table VENUE avec la distribution ALL.

```
create table venue(
```

```
venueid smallint not null,
venuename varchar(100),
venuecity varchar(30),
venuestate char(2),
venueseats integer,
primary key(venueid))
diststyle all;
```

Créer une table avec la distribution EVEN

L'exemple suivant crée la table MYEVENT à trois colonnes.

```
create table myevent(
eventid int,
eventname varchar(200),
eventcity varchar(30))
diststyle even;
```

La table est distribuée uniformément et n'est pas triée. La table ne comporte aucune colonne DISTKEY ou SORTKEY déclarée.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'myevent';

  column   |          type          | encoding | distkey | sortkey
-----------+------------------------+----------+---------+---------
 eventid   | integer                | lzo      | f       |       0
 eventname | character varying(200) | lzo      | f       |       0
 eventcity | character varying(30)  | lzo      | f       |       0
(3 rows)
```

Créer une table temporaire semblable (LIKE) à une autre table

L'exemple suivant crée une table temporaire nommée TEMPEVENT, qui hérite ses colonnes de la table EVENT.

```
create temp table tempevent(like event);
```

Cette table hérite également les attributs DISTKEY et SORTKEY de sa table parent :

```
select "column", type, encoding, distkey, sortkey
 from pg_table_def where tablename = 'tempevent';
```

```
  column   |            type             | encoding | distkey | sortkey
-----------+-----------------------------+----------+---------+---------
 eventid   | integer                     | none     | t       |       1
 venueid   | smallint                    | none     | f       |       0
 catid     | smallint                    | none     | f       |       0
 dateid    | smallint                    | none     | f       |       0
 eventname | character varying(200)      | lzo      | f       |       0
 starttime | timestamp without time zone | bytedict | f       |       0
(6 rows)
```

Créer une table avec une colonne IDENTITY

L'exemple suivant crée une table nommée VENUE_IDENT, avec une colonne IDENTITY nommée VENUEID. Cette colonne commence par 0 et s'incrémente de 1 à chaque enregistrement. VENUEID est également déclarée comme clé primaire de la table.

```
create table venue_ident(venueid bigint identity(0, 1),
venuename varchar(100),
venuecity varchar(30),
venuestate char(2),
venueseats integer,
primary key(venueid));
```

Créer une table avec une colonne IDENTITY par défaut

L'exemple suivant crée une table nommée `t1`. Cette table a une colonne IDENTITY nommée `hist_id` et une colonne IDENTITY par défaut nommée `base_id`.

```
CREATE TABLE t1(
  hist_id BIGINT IDENTITY NOT NULL, /* Cannot be overridden */
  base_id BIGINT GENERATED BY DEFAULT AS IDENTITY NOT NULL, /* Can be overridden */
  business_key varchar(10) ,
  some_field varchar(10)
);
```

L'insertion d'une ligne dans la table montre que les valeurs `hist_id` et `base_id` sont générées.

```
INSERT INTO T1 (business_key, some_field) values ('A','MM');
```

```
SELECT * FROM t1;
```

```
 hist_id | base_id | business_key | some_field
---------+---------+--------------+------------
       1 |       1 | A            | MM
```

L'insertion d'une deuxième ligne montre que la valeur par défaut pour `base_id` est générée.

```
INSERT INTO T1 (base_id, business_key, some_field) values (DEFAULT, 'B','MNOP');
```

```
SELECT * FROM t1;

 hist_id | base_id | business_key | some_field
---------+---------+--------------+------------
       1 |       1 | A            | MM
       2 |       2 | B            | MNOP
```

L'insertion d'une troisième ligne montre que la valeur pour `base_id` n'a pas besoin d'être unique.

```
INSERT INTO T1 (base_id, business_key, some_field) values (2,'B','MNNN');
```

```
SELECT * FROM t1;

 hist_id | base_id | business_key | some_field
---------+---------+--------------+------------
       1 |       1 | A            | MM
       2 |       2 | B            | MNOP
       3 |       2 | B            | MNNN
```

Créer une table avec les valeurs de colonne DEFAULT

L'exemple suivant crée une table CATEGORYDEF qui déclare les valeurs par défaut de chaque colonne :

```
create table categorydef(
catid smallint not null default 0,
catgroup varchar(10) default 'Special',
catname varchar(10) default 'Other',
catdesc varchar(50) default 'Special events',
primary key(catid));
```

```
insert into categorydef values(default,default,default,default);
```

```
select * from categorydef;

 catid | catgroup | catname |    catdesc
-------+----------+---------+----------------
     0 | Special  | Other   | Special events
(1 row)
```

Options DISTSTYLE, DISTKEY et SORTKEY

L'exemple suivant montre comment fonctionnent les options DISTKEY, SORTKEY et DISTSTYLE. Dans cet exemple, COL1 c'est la clé de distribution ; par conséquent, le style de distribution doit être défini sur KEY ou non défini. Par défaut, comme la table n'a aucune clé de tri, elle n'est pas triée :

```
create table t1(col1 int distkey, col2 int) diststyle key;
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't1';

column |  type   | encoding | distkey | sortkey
-------+---------+----------+---------+---------
col1   | integer | az64     | t       | 0
col2   | integer | az64     | f       | 0
```

Dans l'exemple suivant, la même colonne est définie comme clé de distribution et clé de tri. Une fois encore, le style de distribution doit être défini sur KEY ou ne pas être défini.

```
create table t2(col1 int distkey sortkey, col2 int);
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't2';

column |  type   | encoding | distkey | sortkey
-------+---------+----------+---------+---------
col1   | integer | none     | t       | 1
col2   | integer | az64     | f       | 0
```

Dans l'exemple suivant, aucune colonne n'est définie comme clé de distribution, COL2 elle est définie comme clé de tri et le style de distribution est défini sur ALL :

```
create table t3(col1 int, col2 int sortkey) diststyle all;
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't3';

Column |  Type   | Encoding | DistKey | SortKey
-------+---------+----------+---------+--------
col1   | integer | az64     | f       | 0
col2   | integer | none     | f       | 1
```

Dans l'exemple suivant, le style de distribution est défini sur EVEN et aucune clé de tri n'est définie explicitement ; par conséquent, la table est distribuée uniformément, mais n'est pas triée.

```
create table t4(col1 int, col2 int) diststyle even;
```

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 't4';

             column |  type   |encoding | distkey | sortkey
--------+---------+---------+---------+--------
col1    | integer | az64    | f       | 0
col2    | integer | az64    | f       | 0
```

Création d'une table à l'aide de l'option ENCODE AUTO

L'exemple suivant crée la table `t1` avec encodage de compression automatique. ENCODE AUTO est la valeur par défaut pour les tables lorsque vous ne spécifiez pas de type d'encodage pour une colonne.

```
create table t1(c0 int, c1 varchar);
```

L'exemple suivant crée la table `t2` avec encodage de compression automatique en spécifiant ENCODE AUTO.

```
create table t2(c0 int, c1 varchar) encode auto;
```

L'exemple suivant crée la table `t3` avec encodage de compression automatique en spécifiant ENCODE AUTO. La colonne `c0` est définie avec un type d'encodage initial DELTA. Amazon Redshift peut modifier l'encodage si un autre encodage fournit de meilleures performances de requête.

```
create table t3(c0 int encode delta, c1 varchar) encode auto;
```

L'exemple suivant crée la table `t4` avec encodage de compression automatique en spécifiant ENCODE AUTO. La colonne `c0` est définie avec un encodage initial de DELTA, et la colonne `c1` est définie avec un encodage initial de LZO. Amazon Redshift peut modifier ces encodages si d'autres encodages fournissent de meilleures performances de requête.

```
create table t4(c0 int encode delta, c1 varchar encode lzo) encode auto;
```

CREATE TABLE AS

Rubriques

- Syntaxe
- Paramètres
- Notes d'utilisation de CTAS
- Exemples de CTAS

Crée une table basée sur une requête. Le propriétaire de cette table est l'utilisateur qui émet la commande.

La nouvelle table est chargée avec les données définies par la requête dans la commande. Les colonnes ont des noms et des types de données associés aux colonnes de sortie de la requête. La commande CREATE TABLE AS (CTAS) crée une table et évalue la requête pour charger la nouvelle table.

Syntaxe

```
CREATE [ [ LOCAL ] { TEMPORARY | TEMP } ]
TABLE table_name
[ ( column_name [, ... ] ) ]
[ BACKUP { YES | NO } ]
[ table_attributes ]
AS query

where table_attributes are:
[ DISTSTYLE { AUTO | EVEN | ALL | KEY } ]
[ DISTKEY( distkey_identifier ) ]
```

```
[ [ COMPOUND | INTERLEAVED ] SORTKEY( column_name [, ...] ) ]
```

Paramètres

LOCAL

Même si ce mot-clé facultatif est accepté dans l'instruction, il n'a aucun effet dans Amazon Redshift.

TEMPORARY | TEMP

Crée une table temporaire. Une table temporaire est automatiquement supprimée à la fin de la séance dans laquelle elle a été créée.

table_name

Nom de la table à créer.

> **Important**
>
> Si vous spécifiez un nom de table qui commence par « # », la table est créée comme table temporaire. Par exemple :
>
> ```
> create table #newtable (id) as select * from oldtable;
> ```

La longueur maximale d'un nom de table est de 127 octets ; les noms plus longs sont tronqués à 127 octets. Amazon Redshift applique un quota correspondant au nombre de tables par cluster et par type de nœud. Le nom de la table peut être qualifié avec le nom de la base de données et le nom du schéma, comme illustré dans le tableau ci-dessous.

```
create table tickit.public.test (c1) as select * from oldtable;
```

Dans cet exemple, `tickit` est le nom de la base de données et `public` le nom du schéma. Si la base de données ou le schéma n'existe pas, l'instruction renvoie une erreur.

Si un nom de schéma est donné, la nouvelle table est créée dans ce schéma (en supposant que le créateur ait accès au schéma). Le nom de la table doit être un nom unique pour ce schéma. Si aucun schéma n'est spécifié, la table est créée à l'aide du schéma de base de données actuel. Si vous créez une table temporaire, vous ne pouvez pas spécifier un nom de schéma, car les tables temporaires existent dans un schéma spécial.

Plusieurs tables temporaires avec le même nom sont autorisées à exister en même temps dans la base de données même si elles sont créées dans des séances distinctes. Ces tables sont affectées à des schémas différents.

column_name

Nom d'une colonne de la nouvelle table. Si aucun nom de la colonne n'est fourni, les noms de colonnes sont extraits des noms de colonnes de sortie de la requête. Les noms de colonne par défaut sont utilisés pour les expressions. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

BACKUP { YES | NO }

Clause qui spécifie si la table doit être incluse dans les instantanés de cluster automatiques et manuels.

Pour les tables, telles que les tables intermédiaires, qui ne contiennent pas de données critiques, spécifiez BACKUP NO pour économiser du temps de traitement lorsque la création d'instantanés et la restauration à partir d'instantanés, et pour réduire l'espace de stockage sur Amazon Simple Storage Service. Comme le paramètre BACKUP NO n'a aucun effet sur la réplication automatique des données sur d'autres nœuds au sein du cluster, les tables pour lesquelles BACKUP NO est spécifié sont restaurées en cas de défaillance de nœud. La valeur par défaut est BACKUP YES.

> **Note**
>
> Les tables sans sauvegarde ne sont pas prises en charge pour RA3 et sans serveur. Une table marquée comme non sauvegardée RA3 et Serverless sera traitée comme une table permanente qui sera toujours sauvegardée lors de la prise d'un instantané et restaurée lors de la restauration à partir d'un instantané.

DISTSTYLE { AUTO | EVEN | KEY | ALL }

Mot-clé qui définit le style de distribution des données pour l'ensemble de la table. Amazon Redshift répartit les lignes d'une table entre les nœuds de calcul selon le style de distribution spécifié pour la table. La valeur par défaut est DISTSTYLE AUTO.

Le style de distribution que vous sélectionnez pour les tables affecte les performances globales de votre base de données. Pour plus d'informations, consultez Distribution des données pour l'optimisation des requêtes.

- AUTO : Amazon Redshift attribue un style de distribution optimal basé sur les données de table. Pour consulter le style de distribution appliqué à une table, interrogez la table catalogue système PG_CLASS. Pour plus d'informations, consultez Affichage des styles de distribution.
- EVEN : les données de la table sont réparties également entre les nœuds d'un cluster dans une distribution en tourniquet (round robin). IDs Les lignes sont utilisées pour déterminer la distribution, et approximativement le même nombre de lignes est distribué à chaque nœud. Il s'agit de la méthode de distribution par défaut.
- KEY : les données sont réparties selon les valeurs de la colonne DISTKEY. Lorsque vous définissez les colonnes de jointure des tables de jointure comme clés de distribution, les lignes de jointure des deux tables sont colocalisées sur les nœuds de calcul. Lorsque les données sont colocalisées, l'optimiseur peut effectuer les jointures plus efficacement. Si vous spécifiez DISTSTYLE KEY, vous devez nommer une colonne DISTKEY.
- ALL : une copie de la table complète est distribuée sur chaque nœud. Ce style de distribution garantit que toutes les lignes obligatoires pour une jointure sont disponibles sur chaque nœud, mais il multiplie les besoins de stockage et augmente les temps de charge et de maintenance de la table. Une distribution ALL peut améliorer le temps d'exécution lorsqu'elle est utilisée avec certaines tables de dimension où la distribution KEY ne convient pas, mais les améliorations des performances doivent être pondérées par rapport aux coûts de maintenance.

DISTKEY (colonne)

Spécifie un nom de la colonne ou un numéro positionnel pour la clé de distribution. Utilisez le nom spécifié dans la liste facultative de colonnes de la table ou de la liste de sélection de la requête. Sinon, utilisez un numéro positionnel, où la première colonne sélectionnée est 1, la deuxième 2 et ainsi de suite. Une seule colonne d'une table peut être la clé de distribution.

- Si vous déclarez une colonne comme colonne DISTKEY, DISTSTYLE doit être défini sur KEY ou ne pas être défini.
- Si vous ne déclarez pas une colonne DISTKEY, vous pouvez définir DISTSTYLE sur EVEN.
- Si vous ne spécifiez pas DISTKEY ou DISTSTYLE, CTAS détermine le style de distribution de la nouvelle table basée sur le plan de requête de la clause SELECT. Pour plus d'informations, consultez Héritage des attributs de colonne et de table.

Vous pouvez définir la même colonne comme clé de distribution et clé de tri ; cette approche a tendance à accélérer les jointures lorsque la colonne en question est une colonne de jointure de la requête.

[COMPOUND | INTERLEAVED] SORTKEY (nom_colonne [, ...])

Spécifie une ou plusieurs clés de tri pour la table. Lorsque les données sont chargées dans la table, les données sont triées sur les colonnes désignées comme clés de tri.

Vous pouvez spécifier le cas échéant le style de tri COMPOUND ou INTERLEAVED. La valeur par défaut est COMPOUND. Pour plus d'informations, consultez Clés de tri.

Vous pouvez définir un maximum de 400 colonnes COMPOUND SORTKEY ou de 8 colonnes INTERLEAVED SORTKEY par table.

Si vous ne spécifiez pas SORTKEY, CTAS détermine les clés de tri de la nouvelle table basées sur le plan de requête de la clause SELECT. Pour plus d'informations, consultez Héritage des attributs de colonne et de table.

COMPOUND

Spécifie que les données sont triées à l'aide d'une clé composée, constituée de toutes les colonnes affichées, dans leur ordre d'apparition. Une clé de tri composée est surtout utile lorsqu'une requête analyse les lignes selon l'ordre des colonnes de tri. Les avantages en termes de performances d'un tri avec une clé composée diminuent lorsque les requêtes reposent sur des colonnes de tri secondaires. Vous pouvez définir un maximum de 400 colonnes COMPOUND SORTKEY par table.

INTERLEAVED

Spécifie que les données sont triées à l'aide d'une clé de tri entrelacée. Un maximum de huit colonnes peut être spécifié pour une clé de tri entrelacée.

Comme un tri entrelacé confère un poids égal à chaque colonne, ou sous-ensemble de colonnes, de la clé de tri, les requêtes ne dépendent pas de l'ordre des colonnes de la clé de tri. Quand une requête utilise une ou plusieurs colonnes de tri secondaires, le tri entrelacé améliore les performances des requêtes de façon significative. Le tri entrelacé entraîne un léger coût de traitement pour les opérations de chargement de données et les opérations VACUUM.

AS requête

N'importe quelle requête (instruction SELECT) qu'Amazon Redshift prend en charge.

Notes d’utilisation de CTAS

Limites

Amazon Redshift applique un quota correspondant au nombre de tables par cluster et par type de nœud.

Le nombre maximal de caractères pour un nom de table est 127.

Le nombre maximal de colonnes que vous pouvez définir dans une seule table est de 1 600.

Héritage des attributs de colonne et de table

Les tables CREATE TABLE AS (CTAS) n’héritent pas des contraintes, des colonnes d’identité, des valeurs de colonne par défaut ou de la clé primaire de la table à partir desquels elles ont été créées.

Vous ne pouvez pas spécifier d’encodage de compression de colonne pour les tables CTAS. Amazon Redshift attribue automatiquement l’encodage de compression comme suit :

- Les colonnes qui sont définies comme des clés de tri se voient attribuer une compression RAW.
- Les colonnes qui sont définies comme des types de données BOOLEAN, REAL, DOUBLE PRECISION ou GEOMETRY ou GEOGRAPHY se voient attribuer une compression RAW.
- Les colonnes définies comme SMALLINT, INTEGER, BIGINT, DECIMAL, DATE, TIME, TIMETZ, TIMESTAMP ou TIMESTAMPTZ sont soumises à une compression. AZ64
- Les colonnes définies comme CHAR, VARCHAR ou VARBYTE sont affectées à la compression LZO.

Pour plus d’informations, consultez encodages de compression et Types de données.

Pour attribuer explicitement des encodages de colonne, utilisez CREATE TABLE.

CTAS détermine le style de distribution et la clé de tri de la nouvelle table basée sur le plan de requête de la clause SELECT.

Pour les requêtes complexes, telles que les requêtes qui incluent les jointures, les regroupements, une clause order by ou une clause limit, CTAS met tout en œuvre pour choisir le style de distribution optimal et la clé de tri basée sur le plan de requête.

> **Note**
>
> Pour de meilleures performances avec de grands ensembles de données ou des requêtes complexes, nous vous recommandons d'effectuer des tests à l'aide d'ensembles de données classiques.

Vous pouvez souvent prévoir la clé de distribution et la clé de tri que CTAS choisit en examinant le plan de requête pour afficher les colonnes que, le cas échéant, l'optimiseur de requête choisit pour le tri et la répartition des données. Si le nœud supérieur du plan de requête est une analyse séquentielle simple d'une seule table (XN Seq Scan), CTAS utilise généralement ensuite le style de distribution et la clé de tri de la table source. Si le nœud supérieur du plan de requête est autre chose qu'un scan séquentiel (tel que XN Limit, XN Sort, XN, etc.) HashAggregate, le CTAS fait de son mieux pour choisir le style de distribution et la clé de tri optimaux en fonction du plan de requête.

Par exemple, supposons que vous créez cinq tables à l'aide des types suivants de clauses SELECT :

- Une instruction select simple
- Une clause limit
- Une clause order by utilisant LISTID
- Une clause order utilisant QTYSOLD
- Une fonction d'agrégation SUM avec une clause group by.

Les exemples suivants illustrent le plan de requête pour chaque instruction CTAS.

```
explain create table sales1_simple as select listid, dateid, qtysold from sales;
                           QUERY PLAN
-----------------------------------------------------------------
 XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(1 row)


explain create table sales2_limit as select listid, dateid, qtysold from sales limit
 100;
                              QUERY PLAN
---------------------------------------------------------------------
 XN Limit  (cost=0.00..1.00 rows=100 width=8)
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(2 rows)
```

```
explain create table sales3_orderbylistid as select listid, dateid, qtysold from sales
 order by listid;
                               QUERY PLAN
-----------------------------------------------------------------------
 XN Sort  (cost=1000000016724.67..1000000017155.81 rows=172456 width=8)
   Sort Key: listid
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(3 rows)


explain create table sales4_orderbyqty as select listid, dateid, qtysold from sales
 order by qtysold;
                               QUERY PLAN
-----------------------------------------------------------------------
 XN Sort  (cost=1000000016724.67..1000000017155.81 rows=172456 width=8)
   Sort Key: qtysold
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(3 rows)


explain create table sales5_groupby as select listid, dateid, sum(qtysold) from sales
 group by listid, dateid;
                               QUERY PLAN
-----------------------------------------------------------------------
 XN HashAggregate  (cost=3017.98..3226.75 rows=83509 width=8)
   ->  XN Seq Scan on sales  (cost=0.00..1724.56 rows=172456 width=8)
(2 rows)
```

Pour afficher la clé de distribution et la clé de tri de chaque table, interrogez la table catalogue système PG_TABLE_DEF, comme illustré ci-après.

```
select * from pg_table_def where tablename like 'sales%';

      tablename       |   column   | distkey | sortkey
----------------------+------------+---------+---------
 sales                | salesid    | f       |       0
 sales                | listid     | t       |       0
 sales                | sellerid   | f       |       0
 sales                | buyerid    | f       |       0
 sales                | eventid    | f       |       0
 sales                | dateid     | f       |       1
```

```
 sales                | qtysold    | f       |       0
 sales                | pricepaid  | f       |       0
 sales                | commission | f       |       0
 sales                | saletime   | f       |       0
 sales1_simple        | listid     | t       |       0
 sales1_simple        | dateid     | f       |       1
 sales1_simple        | qtysold    | f       |       0
 sales2_limit         | listid     | f       |       0
 sales2_limit         | dateid     | f       |       0
 sales2_limit         | qtysold    | f       |       0
 sales3_orderbylistid | listid     | t       |       1
 sales3_orderbylistid | dateid     | f       |       0
 sales3_orderbylistid | qtysold    | f       |       0
 sales4_orderbyqty    | listid     | t       |       0
 sales4_orderbyqty    | dateid     | f       |       0
 sales4_orderbyqty    | qtysold    | f       |       1
 sales5_groupby       | listid     | f       |       0
 sales5_groupby       | dateid     | f       |       0
 sales5_groupby       | sum        | f       |       0
```

Le tableau suivant résume les résultats. Pour simplifier, nous choisissons d'ignorer les détails des coûts, des lignes et de la largeur du plan d'explication.

Tableau	Instruction select CTAS	Nœud supérieur du plan d'explication	Clé de distribution	Clé de tri
S1_SIMPLE	`select listid, dateid, qtysold from sales`	`XN Seq Scan on sales ...`	LISTID	DATEID
S2_LIMIT	`select listid, dateid, qtysold from sales limit 100`	`XN Limit ...`	Aucun (EVEN)	Aucun
S3_ORDER_BY_LISTID	`select listid, dateid, qtysold from sales order by listid`	`XN Sort ...` `Sort Key: listid`	LISTID	LISTID

Tableau	Instruction select CTAS	Nœud supérieur du plan d'explication	Clé de distribution	Clé de tri
S4_ORDER_BY_QTY	`select listid, dateid, qtysold from sales order by qtysold`	`XN Sort ...` `Sort Key: qtysold`	LISTID	QTYSOLD
S5_GROUP_BY	`select listid, dateid, sum(qtysold) from sales group by listid, dateid`	`XN HashAggregate ...`	Aucun (EVEN)	Aucun

Vous pouvez spécifier explicitement le style de distribution et la clé de tri dans l'instruction CTAS. Par exemple, l'instruction suivante crée une table à l'aide de la distribution EVEN et spécifie SALESID en tant que clé de tri.

```
create table sales_diseven
diststyle even
sortkey (salesid)
as
select eventid, venueid, dateid, eventname
from event;
```

Encodage de compression

ENCODE AUTO est utilisé comme valeur par défaut pour les tables. Amazon Redshift gère automatiquement l'encodage de compression pour toutes les colonnes de la table.

Distribution des données entrantes

Lorsque le schéma de distribution de hachage des données entrantes correspond à celui de la table cible, aucune distribution physique des données n'est réellement nécessaire lors du chargement des données. Par exemple, si une clé de distribution est définie pour la nouvelle table et que les données sont insérées à partir d'une autre table qui est distribuée sur la même colonne de clé, les données sont chargées en place, à l'aide des mêmes nœuds et tranches. Cependant, si les tables source et

cible sont toutes deux définies sur la distribution EVEN, les données sont redistribuées dans la table cible.

Opérations ANALYZE automatiques

Amazon Redshift analyse automatiquement les tables que vous créez avec les commandes CTAS. Vous n'avez pas besoin d'exécuter la commande ANALYZE sur ces tables lorsqu'elles sont créées initialement. Si vous les modifiez, vous devez les analyser de la même manière que les autres tables.

Exemples de CTAS

L'exemple suivant crée une table appelée EVENT_BACKUP pour la table EVENT :

```
create table event_backup as select * from event;
```

La table résultante hérite les clés de distribution et les clés de tri de la table EVENT.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'event_backup';

column    | type                        | encoding | distkey | sortkey
----------+-----------------------------+----------+---------+--------
catid     | smallint                    | none     | false   |       0
dateid    | smallint                    | none     | false   |       1
eventid   | integer                     | none     | true    |       0
eventname | character varying(200)      | none     | false   |       0
starttime | timestamp without time zone | none     | false   |       0
venueid   | smallint                    | none     | false   |       0
```

La commande suivante crée une table appelée EVENTDISTSORT en sélectionnant quatre colonnes de la table EVENT. La nouvelle table est distribuée sur EVENTID et triée sur EVENTID et DATEID :

```
create table eventdistsort
distkey (1)
sortkey (1,3)
as
select eventid, venueid, dateid, eventname
from event;
```

Le résultat est le suivant :

```
select "column", type, encoding, distkey, sortkey
```

```
from pg_table_def where tablename = 'eventdistsort';

column   |          type          | encoding | distkey | sortkey
---------+------------------------+----------+---------+-------
eventid   | integer                | none     | t       | 1
venueid   | smallint               | none     | f       | 0
dateid    | smallint               | none     | f       | 2
eventname | character varying(200)| none     | f       | 0
```

Vous pouvez créer exactement la même table à l'aide des noms de colonne pour les clés de tri et de distribution. Par exemple :

```
create table eventdistsort1
distkey (eventid)
sortkey (eventid, dateid)
as
select eventid, venueid, dateid, eventname
from event;
```

L'instruction suivante applique une distribution uniforme à la table, mais ne définit pas une clé de tri explicite :

```
create table eventdisteven
diststyle even
as
select eventid, venueid, dateid, eventname
from event;
```

La table n'hérite pas de la clé de tri de la table EVENT (EVENTID), car la distribution EVEN a été spécifiée pour la nouvelle table. La nouvelle table n'a ni clé de tri, ni clé de distribution.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'eventdisteven';

column    |          type          | encoding | distkey | sortkey
----------+------------------------+----------+---------+---------
eventid   | integer                | none     | f       | 0
venueid   | smallint               | none     | f       | 0
dateid    | smallint               | none     | f       | 0
eventname | character varying(200) | none     | f       | 0
```

L'instruction suivante applique une distribution uniforme et définit une clé de tri :

```
create table eventdistevensort diststyle even sortkey (venueid)
as select eventid, venueid, dateid, eventname from event;
```

La table en résultant dispose d'une clé de tri, mais pas de clé de distribution.

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'eventdistevensort';

column    |          type          | encoding | distkey | sortkey
----------+------------------------+----------+---------+-------
eventid   | integer                | none     | f       | 0
venueid   | smallint               | none     | f       | 1
dateid    | smallint               | none     | f       | 0
eventname | character varying(200) | none     | f       | 0
```

L'instruction suivante répartit la table EVENT sur une autre colonne de clé à partir des données entrantes, qui sont triées sur la colonne EVENTID, et ne définit aucune colonne SORTKEY ; par conséquent, la table n'est pas triée.

```
create table venuedistevent distkey(venueid)
as select * from event;
```

Le résultat est le suivant :

```
select "column", type, encoding, distkey, sortkey
from pg_table_def where tablename = 'venuedistevent';

 column   |            type             | encoding | distkey | sortkey
----------+-----------------------------+----------+---------+-------
eventid   | integer                     | none     | f       | 0
venueid   | smallint                    | none     | t       | 0
catid     | smallint                    | none     | f       | 0
dateid    | smallint                    | none     | f       | 0
eventname | character varying(200)      | none     | f       | 0
starttime | timestamp without time zone | none     | f       | 0
```

CREATE USER

Crée un nouvel utilisateur de base de données. Les utilisateurs de la base de données peuvent extraire des données, exécuter des commandes et effectuer d'autres actions dans une base de données, en fonction de leurs privilèges et de leurs rôles. Vous devez être un super-utilisateur de la base de données pour exécuter cette commande.

Privilèges requis

Les privilèges suivants sont requis pour CREATE USER :

- Superuser
- Utilisateurs disposant du privilège CREATE USER

Syntaxe

```
CREATE USER name [ WITH ]
PASSWORD { 'password' | 'md5hash' | 'sha256hash' | DISABLE }
[ option [ ... ] ]

where option can be:

CREATEDB | NOCREATEDB
| CREATEUSER | NOCREATEUSER
| SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }
| IN GROUP groupname [, ... ]
| VALID UNTIL 'abstime'
| CONNECTION LIMIT { limit | UNLIMITED }
| SESSION TIMEOUT limit
| EXTERNALID external_id
```

Paramètres

nom

Nom de l'utilisateur créer. Le nom d'utilisateur ne peut pas être `PUBLIC`. Pour plus d'informations sur les noms valides, consultez Noms et identificateurs.

WITH

Mot-clé facultatif. WITH est ignoré par Amazon Redshift

PASSWORD { ’password’ | ’md5hash’ | ’sha256hash’ | DISABLE }

Définit le mot de passe de l’utilisateur.

Par défaut, les utilisateurs peuvent modifier leurs propres mots de passe, sauf si le mot de passe est désactivé. Pour désactiver le mot de passe d’un utilisateur, spécifiez DISABLE. Lorsque le mot de passe d'un utilisateur est désactivé, le mot de passe est supprimé du système et l'utilisateur ne peut se connecter qu'à l'aide d'informations d'identification utilisateur temporaires AWS Identity and Access Management (IAM). Pour plus d’informations, consultez Utilisation de l’authentification IAM pour générer des informations d’identification de l’utilisateur de base de données. Seul un super-utilisateur peut activer ou désactiver des mots de passe. Vous ne pouvez pas désactiver le mot de passe d’un super-utilisateur. Pour activer un mot de passe, exécutez ALTER USER et spécifiez un mot de passe.

Vous pouvez spécifier le mot de passe en texte clair, sous forme de chaîne de MD5 hachage ou de chaîne de SHA256 hachage.

> **Note**
>
> Lorsque vous lancez un nouveau cluster à l'aide de l'API AWS Management Console AWS CLI, ou Amazon Redshift, vous devez fournir un mot de passe en texte clair pour l'utilisateur initial de la base de données. Vous pourrez changer le mot de passe plus tard en utilisant ALTER USER.

Pour le texte clair, le mot de passe doit respecter les contraintes suivantes :

- Sa longueur doit être comprise entre 8 et 64 caractères.
- Il doit contenir au moins une lettre majuscule, une lettre minuscule et un nombre.
- Il peut s’agir de n’importe quel caractère ASCII avec les codes ASCII 33 à 126, à l’exception des guillemets simples (’), des guillemets doubles ("), de \, / ou @.

Comme alternative plus sûre à la transmission du paramètre de mot de passe CREATE USER sous forme de texte clair, vous pouvez spécifier le MD5 hachage d'une chaîne incluant le mot de passe et le nom d'utilisateur.

> **Note**
>
> Lorsque vous spécifiez une chaîne de MD5 hachage, la commande CREATE USER vérifie s'il s'agit d'une chaîne de MD5 hachage valide, mais elle ne valide pas la partie mot

de passe de la chaîne. Dans ce cas, il est possible de créer un mot de passe, tel qu'une chaîne vide, que vous ne pouvez pas utiliser pour vous connecter à la base de données.

Pour définir un MD5 mot de passe, procédez comme suit :

1. Concaténez le nom d'utilisateur et le mot de passe.

   Par exemple, pour le mot de passe `ez` et l'utilisateur `user1`, la chaîne concaténée est `ezuser1`.
2. Convertissez la chaîne concaténée en une chaîne de hachage de 32 MD5 caractères. Vous pouvez utiliser n'importe quel MD5 utilitaire pour créer la chaîne de hachage. L'exemple suivant utilise Amazon Redshift MD5 fonction et l'opérateur de concaténation (||) pour renvoyer une chaîne de hachage de 32 caractères. MD5

   ```
   select md5('ez' || 'user1');

   md5
   ----------------------------------
   153c434b4b77c89e6b94f12c5393af5b
   ```
3. Concaténez « *md5* » devant la chaîne de hachage et fournissez la chaîne concaténée comme argument MD5 md5hash.

   ```
   create user user1 password 'md5153c434b4b77c89e6b94f12c5393af5b';
   ```
4. Connectez-vous à la base de données à l'aide des informations d'identification.

   Pour cet exemple, connectez-vous en tant que `user1` avec le mot de passe `ez`.

Une autre alternative sécurisée consiste à spécifier un hachage SHA-256 d'une chaîne de mots de passe. Vous pouvez également fournir votre propre digest SHA-256 et un salt de 256 bits valides utilisés pour créer le digest.

- Digest : résultat d'une fonction de hachage.
- Salt : données générées de manière aléatoire combinées au mot de passe pour aider à réduire les modèles dans le résultat de la fonction de hachage.

```
'sha256|Mypassword'
```

```
'sha256|digest|256-bit-salt'
```

Dans l'exemple suivant, Amazon Redshift génère et gère le salt.

```
CREATE USER admin PASSWORD 'sha256|Mypassword1';
```

Dans l'exemple suivant, un digest SHA-256 valide et un salt de 256 bits utilisés pour créer le digest sont fournis.

Pour spécifier un mot de passe et le hacher avec votre propre salt, procédez comme suit :

1. Créez un salt de 256 bits. Vous pouvez obtenir un salt en utilisant n'importe quel générateur de chaînes hexadécimales pour générer une chaîne de 64 caractères. Pour cet exemple, le salt est `c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f9513038b0f6c6`.
2. Utilisez la fonction FROM_HEX pour convertir votre salt en binaire. Cela est dû au fait que la SHA2 fonction nécessite la représentation binaire du sel. Observez l'instruction suivante.

   ```
   SELECT
    FROM_HEX('c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f9513038b0f6c6');
   ```
3. Utilisez la fonction CONCAT pour ajouter votre salt à votre mot de passe. Pour cet exemple, le mot de passe est `Mypassword1`. Observez l'instruction suivante.

   ```
   SELECT
    CONCAT('Mypassword1',FROM_HEX('c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f951303
   ```
4. Utilisez cette SHA2 fonction pour créer un résumé à partir de votre combinaison de mot de passe et de sel. Observez l'instruction suivante.

   ```
   SELECT
    SHA2(CONCAT('Mypassword1',FROM_HEX('c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f95
    0);
   ```
5. À l'aide du récapitulatif et du salt des étapes précédentes, créez l'utilisateur. Observez l'instruction suivante.

   ```
   CREATE USER admin PASSWORD 'sha256|
   821708135fcc42eb3afda85286dee0ed15c2c461d000291609f77eb113073ec2|
   c721bff5d9042cf541ff7b9d48fa8a6e545c19a763e3710151f9513038b0f6c6';
   ```

6. Connectez-vous à la base de données à l'aide des informations d'identification.

   Pour cet exemple, connectez-vous en tant que `admin` avec le mot de passe `Mypassword1`.

Si vous définissez un mot de passe en texte brut sans spécifier la fonction de hachage, un MD5 résumé est généré en utilisant le nom d'utilisateur comme sel.

CREATEDB | NOCREATEDB

L'option CREATEDB permet au nouvel utilisateur de créer des bases de données. La valeur par défaut est NOCREATEDB.

CREATEUSER | NOCREATEUSER

L'option CREATEUSER crée un super-utilisateur avec tous les privilèges de base de données, y compris CREATE USER. La valeur par défaut est NOCREATEUSER. Pour plus d'informations, consultez superuser.

SYSLOG ACCESS { RESTRICTED | UNRESTRICTED }

Clause qui spécifie le niveau d'accès de l'utilisateur sur les tableaux système et les vues Amazon Redshift.

Les utilisateurs ordinaires qui disposent de l'autorisation SYSLOG ACCESS RESTRICTED ne peuvent voir que les lignes générées par cet utilisateur dans les tables et les vues système visibles par l'utilisateur. La valeur par défaut est RESTRICTED.

Les utilisateurs ordinaires qui disposent de l'autorisation SYSLOG ACCESS UNRESTRICTED peuvent voir toutes les lignes des tables et des vues système visibles par l'utilisateur, y compris les lignes générées par un autre utilisateur. UNRESTRICTED ne permet pas à un utilisateur standard d'avoir accès aux tableaux visibles du super-utilisateur. Seuls les super-utilisateurs peuvent afficher les données des tableaux visibles des super-utilisateurs.

> **Note**
>
> Accorder à un utilisateur un accès illimité aux tableaux système permet à celui-ci de voir les données générées par d'autres utilisateurs. Par exemple, STL_QUERY et STL_QUERYTEXT contiennent le texte complet des instructions INSERT, UPDATE et DELETE, qui peuvent contenir des données confidentielles générées par l'utilisateur.

Toutes les lignes de SVV_TRANSACTIONS sont visibles de tous les utilisateurs.

Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

IN GROUP nom_groupe

Spécifie le nom d'un groupe existant auquel l'utilisateur appartient. Plusieurs noms de groupe peuvent être répertoriés.

VALID UNTIL heure_absolue

L'option VALID UNTIL définit une heure absolue au-delà de laquelle le mot de passe de l'utilisateur n'est plus valide. Par défaut, le mot de passe n'a pas de limite de temps.

CONNECTION LIMIT { limite | UNLIMITED }

Le nombre maximum de connexions à la base de données que l'utilisateur est autorisé à ouvrir simultanément. La limite se s'applique pas aux super-utilisateurs. Utilisez le mot-clé UNLIMITED pour autoriser le nombre maximum de connexions simultanées. Une limite sur le nombre de connexions pour chaque base de données peut également s'appliquer. Pour plus d'informations, consultez CREATE DATABASE. La valeur par défaut est UNLIMITED. Pour afficher les connexions en cours, interrogez la vue système STV_SESSIONS.

> **Note**
>
> Si les deux limites de connexion (utilisateurs et base de données) s'appliquent, un emplacement de connexion inutilisé situé entre les deux limites doit également être disponible lorsqu'un utilisateur tente de se connecter.

SESSION TIMEOUT limit

Durée maximale en secondes pendant laquelle une séance reste inactive ou en veille. La plage est comprise entre 60 secondes (une minute) et 1 728 000 secondes (20 jours). Si aucun délai d'expiration de séance n'est défini pour l'utilisateur, le paramètre de cluster s'applique. Pour plus d'informations, consultez Quotas et limites dans Amazon Redshift dans le Guide de gestion Amazon Redshift.

Lorsque vous définissez le délai d'expiration de séance, il s'applique uniquement aux nouvelles séances.

Pour afficher des informations sur les séances utilisateur actives, y compris l'heure de début, le nom d'utilisateur et le délai d'expiration de la séance, interrogez la vue système STV_SESSIONS.

Pour afficher des informations sur l'historique des séances utilisateur, interrogez la vue STL_SESSIONS. Pour récupérer des informations sur les utilisateurs de la base de données, y compris les valeurs de délai d'expiration de séance, interrogez la vue SVL_USER_INFO.

EXTERNALID external_id

Identificateur de l'utilisateur associé à un fournisseur d'identité. Le mot de passe de l'utilisateur doit être désactivé. Pour plus d'informations, consultez Fédération de fournisseur d'identité natif pour Amazon Redshift.

Notes d'utilisation

Par défaut, tous les utilisateurs ont les privilèges CREATE et USAGE sur le schéma PUBLIC. Pour interdire aux utilisateurs de créer des objets dans le schéma PUBLIC d'une base de données, utilisez la commande REVOKE pour supprimer ce privilège.

Lorsque vous utilisez l'authentification IAM pour créer des informations d'identification de l'utilisateur de base de données, il se peut que vous vouliez créer un super-utilisateur capable de se connecter en utilisant seulement des informations d'identification temporaires. Vous ne pouvez pas désactiver le mot de passe d'un superutilisateur, mais vous pouvez créer un mot de passe inconnu à l'aide d'une chaîne de MD5 hachage générée aléatoirement.

```
create user iam_superuser password 'md5A1234567890123456780123456789012' createuser;
```

Le cas d'un nom d'utilisateur entre guillemets doubles est toujours conservé quel que soit le réglage de l'option de configuration `enable_case_sensitive_identifier`. Pour plus d'informations, consultez enable_case_sensitive_identifier.

Exemples

La commande suivante crée un utilisateur nommé dbuser, avec le mot de passe « abcD1234 », des privilèges de création de base de données et un nombre de connexions limité à 30.

```
create user dbuser with password 'abcD1234' createdb connection limit 30;
```

Interrogez la table de catalogue PG_USER_INFO pour afficher les détails relatifs à un utilisateur de base de données.

```
select * from pg_user_info;
```

```
 usename   | usesysid | usecreatedb | usesuper | usecatupd | passwd   | valuntil |
 useconfig | useconnlimit
-----------+----------+-------------+----------+-----------+----------+----------
+-----------+-------------
 rdsdb     |        1 | true        | true     | true      | ******** | infinity |
      |
 adminuser |      100 | true        | true     | false     | ******** |          |
      | UNLIMITED
 dbuser    |      102 | true        | false    | false     | ******** |          |
      | 30
```

Dans l'exemple suivant, le mot de passe du compte est valide jusqu'au 10 juin 2017.

```
create user dbuser with password 'abcD1234' valid until '2017-06-10';
```

L'exemple suivant crée un utilisateur avec un mot de passe sensible à la casse qui contient des caractères spéciaux.

```
create user newman with password '@AbC4321!';
```

Pour utiliser une barre oblique inverse (« \ ») dans votre MD5 mot de passe, évitez la barre oblique inverse en ajoutant une barre oblique inverse dans votre chaîne source. L'exemple suivant crée un utilisateur nommé `slashpass` avec une seule barre oblique inverse («\») comme mot de passe.

```
select md5('\\'||'slashpass');

md5
--------------------------------
0c983d1a624280812631c5389e60d48c
```

Créez un utilisateur avec le mot de passe md5.

```
create user slashpass password 'md50c983d1a624280812631c5389e60d48c';
```

L'exemple suivant crée un utilisateur nommé `dbuser` avec un délai d'inactivité de la séance défini sur 120 secondes.

```
CREATE USER dbuser password 'abcD1234' SESSION TIMEOUT 120;
```

L'exemple suivant crée un utilisateur nommé bob. L'espace de noms est myco_aad. Il s'agit uniquement d'un exemple. Pour exécuter la commande avec succès, vous devez disposer d'un fournisseur d'identité enregistré. Pour plus d'informations, consultez Fédération de fournisseur d'identité natif pour Amazon Redshift.

```
CREATE USER myco_aad:bob EXTERNALID "ABC123" PASSWORD DISABLE;
```

CREATE VIEW

Crée une vue dans une base de données. La vue n'est pas physiquement matérialisée ; la requête qui définit la vue est exécutée chaque fois que la vue est référencée dans une requête. Pour créer une vue avec une table externe, incluez la clause WITH NO SCHEMA BINDING.

Pour créer une vue standard, vous avez besoin d'un accès aux tables sous-jacentes ou à des vues sous-jacentes. Pour interroger une vue standard, vous devez sélectionner des autorisations pour la vue elle-même, mais vous ne devez pas en sélectionner pour les tables sous-jacentes. Dans le cas où vous créez une vue qui fait référence à une table ou à une vue d'un autre schéma, ou si vous créez une vue qui fait référence à une vue matérialisée, vous aurez peut-être besoin d'autorisations d'utilisation. Pour interroger une vue à liaison tardive, vous devez sélectionner des autorisations pour la vue elle-même. Vous devez également vous assurer que le propriétaire de la vue de liaison tardive dispose d'autorisations spécifiques pour les objets référencés (tables, vues ou fonctions définies par l'utilisateur). Pour plus d'informations sur les vues à liaison tardive, consultez Notes d'utilisation.

Autorisations requises

Pour utiliser CREATE VIEW, l'une des autorisations suivantes est requise.

- Pour créer une vue à l'aide de CREATE [OR REPLACE] VIEW :
  - Superuser
  - Utilisateurs disposant de l'autorisation CREATE [REPLACE] VIEW
- Pour remplacer une vue existante à l'aide de CREATE OR REPLACE VIEW :
  - Superuser
  - Utilisateurs disposant de l'autorisation CREATE [OR REPLACE] VIEW
  - Propriétaire de la vue

Si un utilisateur souhaite accéder à une vue qui intègre une fonction définie par l'utilisateur, il doit disposer de l'autorisation EXECUTE sur cette fonction.

Syntaxe

```
CREATE [ OR REPLACE ] VIEW name [ ( column_name [, ...] ) ] AS query
[ WITH NO SCHEMA BINDING ]
```

Paramètres

OR REPLACE

Si une vue du même nom existe déjà, la vue est remplacée. Vous pouvez remplacer uniquement par une nouvelle requête qui génère le même ensemble de colonnes, à l'aide des mêmes noms de colonne et des mêmes types de données. CREATE OR REPLACE VIEW verrouille l'affichage pour les lectures et les écritures jusqu'à la fin de l'opération.

Lorsqu'une vue est remplacée, ses autres propriétés telles que la propriété et les privilèges accordés sont conservées.

nom

Nom de la vue. Si un nom de schéma est fourni (tel que `myschema.myview`), la vue est créée à l'aide du schéma spécifié. Sinon, la vue est créée dans le schéma en cours. Le nom de la vue doit être différent du nom de toute autre vue ou table du même schéma.

Si vous spécifiez un nom de vue commençant par « # », la vue est créée comme vue temporaire qui n'est visible que dans la séance en cours.

Pour plus d'informations sur les noms valides, consultez Noms et identificateurs. Vous ne pouvez pas créer de tables ni de vues dans les bases de données système template0, template1, padb_harvest ou sys:internal.

column_name

Liste facultative des noms à utiliser pour les colonnes de la vue. Si aucun nom de colonne n'est fourni, les noms de colonnes sont dérivés de la requête. Le nombre maximal de colonnes que vous pouvez définir dans une seule vue est de 1 600.

query

Requête (sous la forme d'une instruction SELECT) qui est analysée comme table. La table définit les colonnes et les lignes de la vue.

WITH NO SCHEMA BINDING

Clause qui spécifie que la vue n'est pas liée aux objets de base de données sous-jacents, tels que les tables et les fonctions définies par l'utilisateur. Par conséquent, il n'y a aucune dépendance entre la vue et les objets auxquels elle fait référence. Vous pouvez créer une vue même si les objets référencés n'existent pas. Parce qu'il n'y a aucune dépendance, vous pouvez supprimer ou modifier un objet référencé sans affecter la vue. Amazon Redshift ne vérifie pas les dépendances tant que la vue n'est pas interrogée. Pour afficher les détails relatifs aux vues à liaison tardive, exécutez la fonction PG_GET_LATE_BINDING_VIEW_COLS.

Lorsque vous incluez la clause WITH NO SCHEMA BINDING, les tables et vues référencées dans l'instruction SELECT doivent être qualifiées par un nom de schéma. Le schéma doit exister lorsque la vue est créée, même si la table référencée n'existe pas. Par exemple, l'instruction suivante renvoie une erreur.

```
create view myevent as select eventname from event
with no schema binding;
```

L'instruction suivante s'exécute avec succès.

```
create view myevent as select eventname from public.event
with no schema binding;
```

> **Note**
>
> Vous ne pouvez pas mettre à jour, insérer ou supprimer à partir d'une vue.

Notes d'utilisation

Vues à liaison tardive

Une vue à liaison tardive ne contrôle pas les objets de base de données sous-jacents, tels que les autres vues, tant que la vue n'est pas interrogée. En conséquence, vous pouvez supprimer ou modifier les objets sous-jacents sans supprimer la vue et la recréer. Si vous supprimez des objets sous-jacents, les requêtes à la vue à liaison tardive échoueront. Si la requête à la vue à liaison tardive fait référence à des colonnes de l'objet sous-jacent qui ne sont pas présentes, la requête échouera.

Si vous supprimez puis recréez une table ou une vue sous-jacente d'une vue à liaison tardive, le nouvel objet est créé avec des autorisations d'accès par défaut. Vous devrez peut-être accorder des autorisations sur les objets sous-jacents aux utilisateurs qui utiliseront la requête.

Pour créer une vue à liaison tardive, incluez la clause WITH NO SCHEMA BINDING. L'exemple suivant crée une vue sans liaison de schéma.

```
create view event_vw as select * from public.event
with no schema binding;
```

```
select * from event_vw limit 1;

eventid | venueid | catid | dateid | eventname     | starttime
--------+---------+-------+--------+---------------+--------------------
      2 |     306 |     8 |   2114 | Boris Godunov | 2008-10-15 20:00:00
```

L'exemple suivant montre que vous pouvez modifier une table sous-jacente sans recréer la vue.

```
alter table event rename column eventname to title;
```

```
select * from event_vw limit 1;

eventid | venueid | catid | dateid | title         | starttime
--------+---------+-------+--------+---------------+--------------------
      2 |     306 |     8 |   2114 | Boris Godunov | 2008-10-15 20:00:00
```

Vous ne pouvez faire référence aux tables externes Amazon Redshift Spectrum que dans une vue à liaison tardive. Une application des vues à liaison tardive consiste à interroger les tables Amazon Redshift et Redshift Spectrum. Par exemple, vous pouvez utiliser la commande UNLOAD pour archiver les données plus anciennes dans Amazon S3. Ensuite, créez une table externe Redshift Spectrum qui fait référence aux données sur Amazon S3 et créez une vue qui interroge les deux tables. L'exemple suivant utilise une clause UNION ALL pour joindre la table SALES Amazon Redshift et la table SPECTRUM.SALES Redshift Spectrum.

```
create view sales_vw as
select * from public.sales
union all
select * from spectrum.sales
```

```
with no schema binding;
```

Pour plus d'informations sur la création de tables externes Redshift Spectrum, y compris la table `SPECTRUM.SALES`, consultez Mise en route avec Amazon Redshift Spectrum.

> **Important**
>
> Lorsque vous créez une vue standard à partir d'une vue de liaison tardive, la définition de la vue standard contient la définition de la vue de liaison tardive au moment où la vue standard a été créée, y compris le propriétaire de la vue de liaison tardive. Si vous apportez une modification à la vue de liaison tardive sous-jacente, ces modifications ne seront pas utilisées dans la vue standard tant que vous n'aurez pas recréé la vue standard. Par conséquent, lorsque la vue standard est interrogée, elle utilise toujours la définition de la vue de liaison tardive et le propriétaire de la vue de liaison tardive pour vérifier les autorisations au moment de la création de cette vue standard.

Pour mettre à jour la vue standard afin qu'elle fasse référence à la dernière définition de la vue à liaison tardive, exécutez CREATE OR REPLACE VIEW avec la définition de vue initiale que vous avez utilisée pour créer la vue standard.

Consultez l'exemple suivant de création d'une vue standard à partir d'une vue à liaison tardive.

```
create view sales_vw_lbv as
select * from public.sales
with no schema binding;

show view sales_vw_lbv;
                           Show View DDL statement
--------------------------------------------------------------------------------
 create view sales_vw_lbv as select * from public.sales with no schema binding;
(1 row)

create view sales_vw as
select * from sales_vw_lbv;

show view sales_vw;
                                                Show View DDL statement
--------------------------------------------------------------------------------------------------
 SELECT sales_vw_lbv.price, sales_vw_lbv."region" FROM (SELECT sales.price,
 sales."region" FROM sales) sales_vw_lbv;
```

```
(1 row)
```

Notez que la vue à liaison tardive, telle qu'indiquée dans l'instruction DDL pour la vue standard, est définie lors de la création de la vue standard et ne sera pas mise à jour avec les modifications que vous apporterez ultérieurement à la vue à liaison tardive.

Exemples

Les exemples de commandes utilisent un ensemble d'objets et de données appelé base de données TICKIT. Pour plus d'informations, consultez Exemple de base de données.

La commande suivante crée une vue nommée myevent à partir d'une table appelée EVENT.

```
create view myevent as select eventname from event
where eventname = 'LeAnn Rimes';
```

La commande suivante crée une vue nommée myuser à partir d'une table appelée USERS.

```
create view myuser as select lastname from users;
```

La commande suivante crée ou remplace une vue nommée myuser à partir d'une table appelée USERS.

```
create or replace view myuser as select lastname from users;
```

L'exemple suivant crée une vue sans liaison de schéma.

```
create view myevent as select eventname from public.event
with no schema binding;
```

DEALLOCATE

Libère une instruction préparée.

Syntaxe

```
DEALLOCATE [PREPARE] plan_name
```

Paramètres

PREPARE

Ce mot-clé est facultatif et est ignoré.

nom_plan

Nom de l'instruction préparée à libérer.

Notes d'utilisation

DEALLOCATE permet de libérer une requête SQL préparée précédemment. Si vous ne libérez pas explicitement une instruction préparée, elle est libérée lorsque la séance en cours se termine. Pour plus d'informations sur les instructions préparées, consultez PREPARE.

consultez aussi

EXECUTE, PREPARE

DECLARE

Définit un nouveau curseur. Utilisez un curseur pour récupérer en même temps quelques lignes du jeu de résultats d'une requête plus grande.

Lorsque la première ligne d'un curseur est extraite, le jeu complet de résultats est matérialisé sur le nœud principal, en mémoire ou sur disque, si nécessaire. En raison de l'impact négatif potentiel de l'utilisation de curseurs avec des ensembles de résultats volumineux sur les performances, nous recommandons autant que possible l'utilisation d'approches alternatives. Pour plus d'informations, consultez Considérations relatives aux performances lors de l'utilisation de curseurs.

Vous devez déclarer un curseur au sein d'un bloc de transaction. Un seul curseur à la fois peut être ouvert par séance.

Pour plus d'informations, consultez FETCH, CLOSE.

Syntaxe

```
DECLARE cursor_name CURSOR FOR query
```

Paramètres

nom_curseur

Nom du nouveau schéma.

query

Instruction SELECT qui remplit le curseur.

Notes d'utilisation sur DECLARE CURSOR

Si votre application cliente utilise une connexion ODBC et que votre requête crée un ensemble de résultats trop volumineux pour contenir en mémoire, vous pouvez diffuser l'ensemble de résultats sur votre application cliente à l'aide d'un curseur. Lorsque vous utilisez un curseur, l'ensemble de résultats est matérialisé sur le nœud principal et le client peut ensuite extraire les résultats de façon incrémentielle.

> **Note**
>
> Pour activer les curseurs dans ODBC pour Microsoft Windows, activez l'option Use Declare/Fetch (Utiliser Declare/Fetch) dans le DSN ODBC que vous utilisez pour Amazon Redshift. Nous recommandons de définir la taille du cache ODBC, à l'aide du champ Cache Size (Taille du cache) de la boîte de dialogue ODBC DSN, avec la valeur 4 000 ou plus sur les clusters à plusieurs nœuds afin de réduire les allers et retours. Sur un cluster à un seul nœud, définissez Cache Size sur 1 000.

En raison de l'impact négatif potentiel de l'utilisation de curseurs sur les performances, nous recommandons d'utiliser les autres approches autant que possible. Pour plus d'informations, consultez Considérations relatives aux performances lors de l'utilisation de curseurs.

Les curseurs Amazon Redshift sont pris en charge avec les limitations suivantes :

- Un seul curseur à la fois peut être ouvert par séance.
- Les curseurs doivent être utilisés au sein d'une transaction (BEGIN … END).
- La taille maximale cumulée du jeu de résultats pour tous les curseurs est limitée en fonction du type de nœud du cluster. Si vous avez besoin de jeux de résultats plus larges, vous pouvez redimensionner avec une configuration de nœud XL ou 8XL.

Pour plus d'informations, consultez Contraintes de curseur.

Contraintes de curseur

Lorsque la première ligne d'un curseur est extraite, le jeu complet de résultats est matérialisé sur le nœud principal. Si le jeu de résultats ne contient pas en mémoire, il est écrit sur le disque en fonction des besoins. Pour protéger l'intégrité du nœud principal, Amazon Redshift applique les contraintes sur la taille de tous les jeux de résultats du curseur, en fonction du type de nœud du cluster.

Le tableau suivant illustre la taille maximale du jeu de résultats pour chaque type de nœud de cluster. Les tailles maximales des jeux de résultats sont exprimés en mégaoctets.

Type de nœud	Ensemble de résultats maximal par cluster (Mo)
Nœuds multiples DC2 Large	192 000
Nœud simple DC2 Large	8 000
DC2 Nœuds multiples 8XL	3 200 000
RA3 Nœuds multiples 16XL	14 400 000
RA3 Nœuds multiples 4XL	3 200 000
RA3 Nœuds multiples XLPLUS	1 000 000
RA3 Nœud unique XLPLUS	64 000
RA3 Nœuds multiples de grande taille	240 000
RA3 Nœud unique de grande taille	8 000
Amazon Redshift sans serveur	150 000

Pour afficher la configuration de curseur active pour un cluster, interrogez la table système STV_CURSOR_CONFIGURATION en tant que super-utilisateur. Pour afficher l'état des curseurs

actifs, interrogez la table système STV_ACTIVE_CURSORS. Seules les lignes des curseurs d'un utilisateur sont visibles de l'utilisateur, mais un super-utilisateur peut afficher tous les curseurs.

Considérations relatives aux performances lors de l'utilisation de curseurs

Comme les curseurs matérialisent la totalité de l'ensemble de résultats sur le nœud principal avant de commencer à renvoyer les résultats au client, l'utilisation de curseurs avec des ensembles de résultats très volumineux peut avoir un impact négatif sur les performances. Nous vous déconseillons vivement l'utilisation de curseurs avec des jeux de résultats très volumineux. Dans certains cas, comme lorsque votre application utilise une connexion ODBC, les curseurs peuvent être la seule solution possible. Si possible, nous recommandons d'utiliser ces alternatives :

- Utilisez UNLOAD pour exporter une grande table. Lorsque vous utilisez UNLOAD, les nœuds de calcul fonctionnent en parallèle pour transférer directement les données vers les fichiers de données sur Amazon Simple Storage Service. Pour plus d'informations, consultez Déchargement de données dans Amazon Redshift.
- Définissez le paramètre JDBC de taille d'extraction dans votre application cliente. Si vous utilisez une connexion JDBC et que vous rencontrez des out-of-memory erreurs côté client, vous pouvez permettre à votre client de récupérer des ensembles de résultats par lots plus petits en définissant le paramètre JDBC fetch size. Pour de plus amples informations, veuillez consulter Définition du paramètre de taille d'extraction JDBC.

Exemples DECLARE CURSOR

L'exemple suivant déclare un curseur nommé LOLLAPALOOZA pour sélectionner les informations sur les ventes pour l'événement Lollapalooza, puis récupère les lignes de l'ensemble de résultats à l'aide du curseur :

```
-- Begin a transaction

begin;

-- Declare a cursor

declare lollapalooza cursor for
select eventname, starttime, pricepaid/qtysold as costperticket, qtysold
from sales, event
where sales.eventid = event.eventid
and eventname='Lollapalooza';
```

```
-- Fetch the first 5 rows in the cursor lollapalooza:

fetch forward 5 from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
--------------+---------------------+---------------+---------
 Lollapalooza | 2008-05-01 19:00:00 |   92.00000000 |       3
 Lollapalooza | 2008-11-15 15:00:00 |  222.00000000 |       2
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       3
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       4
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       1
(5 rows)

-- Fetch the next row:

fetch next from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
--------------+---------------------+---------------+---------
 Lollapalooza | 2008-10-06 14:00:00 |  114.00000000 |       2

-- Close the cursor and end the transaction:

close lollapalooza;
commit;
```

L’exemple suivant passe en boucle sur un curseur contenant tous les résultats d’une table :

```
CREATE TABLE tbl_1 (a int, b int);
INSERT INTO tbl_1 values (1, 2),(3, 4);

CREATE OR REPLACE PROCEDURE sp_cursor_loop() AS $$
DECLARE
    target record;
    curs1 cursor for select * from tbl_1;
BEGIN
    OPEN curs1;
    LOOP
        fetch curs1 into target;
        exit when not found;
        RAISE INFO 'a %', target.a;
    END LOOP;
```

```
    CLOSE curs1;
END;
$$ LANGUAGE plpgsql;

CALL sp_cursor_loop();

SELECT message
   from svl_stored_proc_messages
   where querytxt like 'CALL sp_cursor_loop()%';

  message
----------
      a 1
      a 3
```

DELETE

Supprime les lignes des tables.

> **Note**
>
> La taille maximale d'une instruction SQL est de 16 Mo.

Syntaxe

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
DELETE [ FROM ] { table_name | materialized_view_name }
    [ { USING } table_name, ... ]
    [ WHERE condition ]
```

Paramètres

Clause WITH

Clause facultative qui en spécifie une ou plusieurs common-table-expressions. Consultez Clause WITH.

FROM

Le mot-clé FROM est facultatif, sauf si la clause USING est spécifiée. Les instructions `delete from event;` et `delete event;` sont des opérations équivalentes qui suppriment toutes les lignes de la table EVENT.

> **Note**
>
> Pour supprimer toutes les lignes d'une table, TRUNCATE la table. TRUNCATE est beaucoup plus efficace que DELETE et ne requiert ni opération VACUUM ni opération ANALYZE. Cependant, sachez que TRUNCATE valide la transaction dans laquelle l'opération est exécutée.

table_name

Table temporaire ou permanente. Seul le propriétaire de la table ou un utilisateur avec le privilège DELETE sur la table peut supprimer des lignes de la table.

Pensez à utiliser la commande TRUNCATE pour les opérations de suppression rapides non qualifiées sur les tables volumineuses ; consultez TRUNCATE.

> **Note**
>
> Après avoir supprimé un grand nombre de lignes d'une table :
>
> - Exécuter une opération VACUUM sur la table pour récupérer de l'espace de stockage et retrier les lignes.
> - Analysez la table pour mettre à jour les statistiques pour le planificateur de requête.

materialized_view_name

Vue matérialisée. L'instruction DELETE fonctionne sur une vue matérialisée utilisée pour Diffusion de l'ingestion vers une vue matérialisée. Seul le propriétaire de la vue matérialisée ou un utilisateur disposant du privilège DELETE sur la vue matérialisée peut y supprimer des lignes.

Vous ne pouvez pas exécuter DELETE sur une vue matérialisée pour l'ingestion en streaming avec une politique de sécurité au niveau des lignes (RLS) qui ne dispose pas de l'autorisation

IGNORE RLS accordée à l'utilisateur. Il existe une exception à cette règle : si RLS IGNORE est accordé à l'utilisateur exécutant DELETE, il s'exécute correctement. Pour plus d'informations, consultez Propriété et gestion des politiques RLS.

USING nom_table, ...

Le mot-clé USING permet d'introduire une liste de tables lorsque des tables supplémentaires sont référencées dans la condition de la clause WHERE. Par exemple, l'instruction suivante supprime toutes les lignes de la table EVENT qui satisfont à la condition de jointure sur les tables EVENT et SALES. La table SALES doit être explicitement nommée dans la liste FROM :

```
delete from event using sales where event.eventid=sales.eventid;
```

Si vous répétez le nom de la table cible dans la clause USING, l'opération DELETE exécute une jointure réflexive. Vous pouvez utiliser une sous-requête dans la clause WHERE au lieu de la syntaxe USING comme alternative à l'écriture de la même requête.

WHERE condition

Clause facultative qui limite la suppression de lignes à celles qui correspondent à la condition. Par exemple, la condition peut être une restriction sur une colonne, une condition de jointure ou une condition basée sur le résultat d'une requête. La requête peut faire référence à des tables autres que la cible de la commande DELETE. Par exemple :

```
delete from t1
where col1 in(select col2 from t2);
```

Si aucune condition n'est spécifiée, toutes les lignes de la table sont supprimées.

Exemples

Supprimez toutes les lignes de la table CATEGORY :

```
delete from category;
```

Supprimez des lignes avec des valeurs CATID comprises entre 0 et 9 de la table CATEGORY :

```
delete from category
```

```
where catid between 0 and 9;
```

Supprimez des lignes de la table LISTING dont les valeurs SELLERID n'existent pas dans la table SALES :

```
delete from listing
where listing.sellerid not in(select sales.sellerid from sales);
```

Les deux requêtes suivantes suppriment une ligne de la table CATEGORY, en fonction d'une jointure avec la table EVENT et d'une restriction supplémentaire sur la colonne CATID :

```
delete from category
using event
where event.catid=category.catid and category.catid=9;
```

```
delete from category
where catid in
(select category.catid from category, event
where category.catid=event.catid and category.catid=9);
```

La requête suivante supprime toutes les lignes de la vue matérialisée `mv_cities`. Le nom de la vue matérialisée ici est un exemple :

```
delete from mv_cities;
```

DESC DATASHARE

Affiche la liste des objets de base de données dans un datashare qui lui sont ajoutés à l'aide de ALTER DATASHARE. Amazon Redshift affiche les noms, les bases de données, les schémas et les types de tables, de vues et de fonctions.

Des informations supplémentaires sur les objets de partage de données peuvent être trouvées en utilisant les vues système. Pour plus d'informations, consultez SVV_DATASHARE_OBJECTS et SVV_DATASHARES.

Syntaxe

```
DESC DATASHARE datashare_name [ OF [ ACCOUNT account_id ] NAMESPACE namespace_guid ]
```

Paramètres

datashare_name

Nom de l’unité de partage des données.

NAMESPACE namespace_guid

Valeur qui spécifie l’espace de noms utilisé par l’unité de partage des données. Lorsque vous exécutez DESC DATAHSARE en tant qu’administrateur de cluster consommateur, spécifiez le paramètre NAMESPACE pour afficher les unités de partage des données entrantes.

ACCOUNT account_id

Valeur spécifiant le compte auquel l’unité de partage des données appartient.

Notes d’utilisation

En tant qu'administrateur de compte client, lorsque vous exécutez DESC DATASHARE pour voir les partages de données entrants au sein du AWS compte, spécifiez l'option NAMESPACE. Lorsque vous exécutez DESC DATASHARE pour voir les partages de données entrants entre les AWS comptes, spécifiez les options ACCOUNT et NAMESPACE.

Exemples

L’exemple suivant affiche les informations relatives aux unités de partage des données sortantes sur un cluster producteur.

```
DESC DATASHARE salesshare;

producer_account |          producer_namespace           | share_type  | share_name   |
 object_type |        object_name           |  include_new
-----------------+--------------------------------------+-------------+--------------
+-------------+------------------------------+--------------
 123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d  | OUTBOUND    |  salesshare  |
 TABLE       | public.tickit_sales_redshift |
 123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d  | OUTBOUND    |  salesshare  |
 SCHEMA      | public                       |   t
```

L’exemple suivant affiche les informations relatives aux unités de partage des données entrantes sur un cluster producteur.

```
DESC DATASHARE salesshare of ACCOUNT '123456789012' NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';

 producer_account |          producer_namespace          | share_type | share_name |
 object_type |         object_name          |  include_new
-----------------+--------------------------------------+------------+------------
+-------------+------------------------------+-------------
 123456789012     | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare |
 table       | public.tickit_sales_redshift |
 123456789012     | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d | INBOUND    | salesshare |
 schema      | public                       |
(2 rows)
```

DESC IDENTITY PROVIDER

Affiche des informations sur un fournisseur d'identité. Seul un super-utilisateur peut décrire un fournisseur d'identité.

Syntaxe

```
DESC IDENTITY PROVIDER identity_provider_name
```

Paramètres

identity_provider_name

Nom du fournisseur d'identité.

exemple

L'exemple suivant affiche des informations sur le fournisseur d'identité.

```
DESC IDENTITY PROVIDER azure_idp;
```

Exemple de sortie.

```
  uid   |   name    | type  |             instanceid             | namespc |
```

```
                                                  params

                           | enabled
--------+----------+-------+--------------------------------------+---------
+-----------------------------------------------------------------------------------------------------
+---------
 126692 | azure_idp | azure | e40d4bb2-7670-44ae-bfb8-5db013221d73 | aad     |
 {"issuer":"https://login.microsoftonline.com/e40d4bb2-7670-44ae-bfb8-5db013221d73/
v2.0", "client_id":"871c010f-5e61-4fb1-83ac-98610a7e9110", "client_secret":'',
 "audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift", "https://
analysis.windows.net/powerbi/connector/AWSRDS"]} | t
(1 row)
```

DETACH MASKING POLICY

Détache une politique de masquage dynamique des données attachée à une colonne. Pour plus d'informations sur le masquage dynamique des données, consultez Masquage dynamique des données.

Les super-utilisateurs et les utilisateurs ou les rôles disposant du rôle sys:secadmin peuvent détacher une politique de masquage.

Syntaxe

```
DETACH MASKING POLICY policy_name
   ON { table_name }
   ( output_column_names )
   FROM { user_name | ROLE role_name | PUBLIC };
```

Paramètres

policy_name

Nom de la politique de masquage à détacher.

table_name

Nom de la table depuis laquelle détacher la politique de masquage.

output_column_names

Noms des colonnes auxquelles la politique de masquage était attachée.

user_name

Nom de l'utilisateur auquel la politique de masquage était attachée.

Vous ne pouvez définir qu'un seul user_name, role_name et PUBLIC dans une seule instruction DETACH MASKING POLICY.

role_name

Nom du rôle auquel la politique de masquage était attachée.

Vous ne pouvez définir qu'un seul user_name, role_name et PUBLIC dans une seule instruction DETACH MASKING POLICY.

PUBLIC

Indique que la politique était attachée à tous les utilisateurs de la table.

Vous ne pouvez définir qu'un seul user_name, role_name et PUBLIC dans une seule instruction DETACH MASKING POLICY.

DETACH RLS POLICY

Détachez une politique de sécurité au niveau des lignes sur une table d'un ou de plusieurs utilisateurs ou rôles.

Les super-utilisateurs et les utilisateurs ou les rôles qui disposent du rôle `sys:secadmin` peuvent détacher une stratégie.

Syntaxe

```
DETACH RLS POLICY policy_name ON [TABLE] table_name [, ...]
FROM { user_name | ROLE role_name | PUBLIC } [, ...]
```

Paramètres

policy_name

Nom de la politique .

ON [TABLE] table_name [, ...]

La table ou la vue à partir de laquelle la politique de sécurité au niveau des lignes est détachée.

FROM { user_name | ROLE role_name | PUBLIC} [, ...]

Spécifie si la politique est détachée à partir d'un ou de plusieurs utilisateurs ou rôles spécifiés.

Notes d'utilisation

Lorsque vous utilisez l'instruction DETACH RLS POLICY, tenez compte des points suivants :

- Vous pouvez détacher une politique d'une relation, d'un utilisateur, d'un rôle ou d'un public.

Exemples

L'exemple suivant détache une politique sur une table à partir d'un rôle.

```
DETACH RLS POLICY policy_concerts ON tickit_category_redshift FROM ROLE analyst, ROLE
 dbadmin;
```

DROP DATABASE

Supprime une base de données.

Vous ne pouvez pas exécuter DROP DATABASE au sein d'un bloc de transaction (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

Syntaxe

```
DROP DATABASE database_name
```

Paramètres

database_name

Nom de la base de données à supprimer. Vous ne pouvez pas supprimer les bases de données dev, padb_harvest, template0, template1 ou sys:internal, et vous ne pouvez pas supprimer la base de données active.

Pour supprimer une base de données externe, supprimez le schéma externe. Pour plus d'informations, consultez DROP SCHEMA.

Notes d'utilisation de DROP DATA

Lors de l'utilisation de l'instruction DROP DATABASE, tenez compte des éléments suivants :

- En règle générale, nous vous recommandons de ne pas supprimer de base de données contenant une unité de partage des données AWS Data Exchange à l'aide de l'instruction DROP DATABASE. Si vous le faites, les Comptes AWS qui ont accès à l'unité de partage des données perdent l'accès. L'exécution de ce type de modification peut enfreindre les termes des produits de données dans AWS Data Exchange.

  L'exemple suivant illustre une erreur lorsqu'une base de données contient une unité de partage des données AWS Data Exchange est supprimée.

  ```
  DROP DATABASE test_db;
  ERROR:   Drop of database test_db that contains ADX-managed datashare(s)
   requires session variable datashare_break_glass_session_var to be set to value
   'ce8d280c10ad41'
  ```

  Pour autoriser la suppression de la base de données, définissez la variable suivante et exécutez à nouveau l'instruction DROP DATABASE.

  ```
  SET datashare_break_glass_session_var to 'ce8d280c10ad41';
  ```

  ```
  DROP DATABASE test_db;
  ```

  Dans ce cas, Amazon Redshift génère une valeur ponctuelle aléatoire pour définir la variable de séance afin d'autoriser DROP DATABASE pour une base de données contenant une unité de partage des données AWS Data Exchange .

Exemples

L'exemple suivant supprime une base de données nommée TICKIT_TEST :

```
drop database tickit_test;
```

DROP DATASHARE

Supprime une unité de partage des données. La commande n'est pas réversible.

Seul un super-utilisateur ou le propriétaire de l'unité de partage des données peut supprimer une unité de partage des données.

Privilèges requis

Les privilèges suivants sont requis pour DROP DATASHARE :

- Superuser
- Utilisateurs disposant du privilège DROP DATASHARE
- Propriétaire de l'unité de partage des données

Syntaxe

```
DROP DATASHARE datashare_name;
```

Paramètres

datashare_name

Nom de l'unité de partage des données à supprimer.

Note d'utilisation de DROP DATASHARE

Lors de l'utilisation de l'instruction DROP DATASHARE, tenez compte des éléments suivants :

- En général, nous vous recommandons de ne pas supprimer un partage de données à l'aide de l'instruction DROP AWS Data Exchange DATASHARE. Dans ce cas, ceux Comptes AWS qui ont accès au partage de données en perdent l'accès. L'exécution de ce type de modification peut enfreindre les termes des produits de données dans AWS Data Exchange.

  L'exemple suivant montre une erreur lors de la suppression d'un AWS Data Exchange partage de données.

  ```
  DROP DATASHARE salesshare;
  ERROR:  Drop of ADX-managed datashare salesshare requires session variable
   datashare_break_glass_session_var to be set to value '620c871f890c49'
  ```

  Pour autoriser la suppression d'un AWS Data Exchange partage de données, définissez la variable suivante et réexécutez l'instruction DROP DATASHARE.

```
SET datashare_break_glass_session_var to '620c871f890c49';
```

```
DROP DATASHARE salesshare;
```

Dans ce cas, Amazon Redshift génère une valeur unique aléatoire pour définir la variable de session afin d'autoriser DROP DATASHARE pour un partage de données. AWS Data Exchange

Exemples

L'exemple suivant supprime une unité de partage des données appelé `salesshare`.

```
DROP DATASHARE salesshare;
```

SUPPRIMER LA VUE EXTÉRIEURE

Supprime une vue externe de la base de données. La suppression d'une vue externe la supprime de tous les moteurs SQL auxquels elle est associée, tels qu'Amazon Athena et Amazon EMR Spark. Cette commande ne peut pas être annulée. Pour plus d'informations sur les vues du catalogue de données, consultez la section AWS Glue Data Catalog vues.

Syntaxe

```
DROP EXTERNAL VIEW schema_name.view_name [ IF EXISTS ]
{catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name |
 external_schema_name.view_name}
```

Paramètres

schema_name.view_name

Le schéma attaché à votre AWS Glue base de données, suivi du nom de la vue.

IF EXISTS

Supprime la vue uniquement si elle existe.

catalog_name.schema_name.view_name | awsdatacatalog.dbname.view_name | external_schema_name.view_name

Notation du schéma à utiliser lors de la suppression de la vue. Vous pouvez spécifier d'utiliser une base de AWS Glue Data Catalog données Glue que vous avez créée ou un schéma externe que vous avez créé. Consultez CREATE DATABASE et CREATE EXTERNAL SCHEMA pour plus d'informations.

query_definition

Définition de la requête SQL exécutée par Amazon Redshift pour modifier la vue.

Exemples

L'exemple suivant supprimer une vue du catalogue de données nommée sample_schema.glue_data_catalog_view.

```
DROP EXTERNAL VIEW sample_schema.glue_data_catalog_view IF EXISTS
```

DROP FUNCTION

Supprime une fonction définie par l'utilisateur (UDF) de la base de données. La signature de la fonction, ou liste des types de données des arguments, doit être spécifiée, car plusieurs fonctions peuvent exister avec le même nom, mais avec des signatures différentes. Vous ne pouvez pas supprimer une fonction intégrée Amazon Redshift.

La commande n'est pas réversible.

Privilèges requis

Les privilèges suivants sont requis pour DROP FUNCTION :

- Superuser
- Utilisateurs disposant du privilège DROP FUNCTION
- Propriétaire de la fonction

Syntaxe

```
DROP FUNCTION name
```

```
( [arg_name] arg_type   [, ...] )
[ CASCADE | RESTRICT ]
```

Paramètres

nom

Nom de la fonction à supprimer.

nom_arg

Nom d'un argument d'entrée. DROP FUNCTION ignore les noms d'argument, car seuls les types de données des arguments sont nécessaires pour déterminer l'identité de la fonction.

type_arg

Type de données de l'argument en entrée. Vous pouvez fournir une liste séparée par des virgules d'un maximum de 32 types de données.

CASCADE

Mot-clé spécifiant de supprimer automatiquement les objets qui dépendent de la fonction, tel que les vues.

Pour créer une vue qui ne dépend pas d'une fonction, incluez la clause WITH NO SCHEMA BINDING dans la définition de vue. Pour plus d'informations, consultez CREATE VIEW.

RESTRICT

Mot-clé spécifiant que si les objets dépendent de la fonction, il ne faut pas supprimer la fonction ni renvoyer un message. Il s'agit de l'action par défaut.

Exemples

L'exemple suivant supprime la fonction nommée `f_sqrt` :

```
drop function f_sqrt(int);
```

Pour supprimer une fonction qui comporte des dépendances, utilisez l'option CASCADE, comme illustré dans l'exemple suivant :

```
drop function f_sqrt(int)cascade;
```

DROP GROUP

Supprime un groupe d'utilisateurs. La commande n'est pas réversible. Elle ne supprime pas les utilisateurs individuels d'un groupe.

Consultez DROP USER pour supprimer un utilisateur.

Syntaxe

```
DROP GROUP name
```

Paramètre

nom

Nom du groupe d'utilisateurs à supprimer.

exemple

L'exemple suivant supprime le groupe `guests` d'utilisateurs :

```
DROP GROUP guests;
```

Vous ne pouvez pas supprimer un groupe si le groupe n'a pas de privilèges sur un objet. Si vous essayez de supprimer un tel groupe, vous recevrez l'erreur suivante.

```
ERROR: group "guests" can't be dropped because the group has a privilege on some object
```

Si le groupe possède des privilèges pour un objet, vous devez révoquer ces privilèges avant de supprimer le groupe. Pour rechercher les objets pour lesquels le `guests` groupe possède des privilèges, utilisez l'exemple suivant. Pour plus d'informations sur la vue des métadonnées utilisée dans l'exemple, consultez SVV_RELATION_PRIVILEGES.

```
SELECT DISTINCT namespace_name, relation_name, identity_name, identity_type
FROM svv_relation_privileges
WHERE identity_type='group' AND identity_name='guests';

+----------------+---------------+---------------+---------------+
| namespace_name | relation_name | identity_name | identity_type |
```

```
+----------------+---------------+---------------+---------------+
| public         | table1        | guests        | group         |
+----------------+---------------+---------------+---------------+
| public         | table2        | guests        | group         |
+----------------+---------------+---------------+---------------+
```

L'exemple suivant révoque tous les privilèges sur toutes les tables du schéma `public` du groupe d'utilisateurs `guests`, puis supprime le groupe.

```
REVOKE ALL ON ALL TABLES IN SCHEMA public FROM GROUP guests;
DROP GROUP guests;
```

DROP IDENTITY PROVIDER

Supprime un fournisseur d'identité. La commande n'est pas réversible. Seul un super-utilisateur peut supprimer un fournisseur d'identité.

Syntaxe

```
DROP IDENTITY PROVIDER identity_provider_name [ CASCADE ]
```

Paramètres

identity_provider_name

Nom du fournisseur d'identité à supprimer.

CASCADE

Supprime les utilisateurs et les rôles attachés au fournisseur d'identité lorsqu'il est supprimé.

exemple

L'exemple suivant supprime le fournisseur d'identité oauth_provider.

```
DROP IDENTITY PROVIDER oauth_provider;
```

Si vous supprimez le fournisseur d'identité, certains utilisateurs peuvent ne pas être en mesure de se connecter ou d'utiliser les outils clients configurés pour utiliser le fournisseur d'identité.

DROP LIBRARY

Supprime une bibliothèque Python personnalisée de la base de données. Seul le propriétaire de la bibliothèque ou un super-utilisateur peut supprimer une bibliothèque.

DROP LIBRARY ne peut pas s'exécuter à l'intérieur d'un bloc de transaction (BEGIN … END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

La commande n'est pas réversible. La commande DROP LIBRARY valide immédiatement. Si une fonction UDF qui dépend de la bibliothèque s'exécute simultanément, la fonction UDF peut échouer, même si la fonction UDF s'exécute au sein d'une transaction.

Pour de plus amples informations, veuillez consulter CREATE LIBRARY.

Privilèges requis

Les privilèges suivants sont requis pour DROP LIBRARY :

- Superuser
- Utilisateurs disposant du privilège DROP LIBRARY
- Propriétaire de la bibliothèque

Syntaxe

```
DROP LIBRARY library_name
```

Paramètres

nom_bibliothèque

Nom de la bibliothèque.

DROP MASKING POLICY

Supprime une politique de masquage dynamique des données de toutes les bases de données. Vous ne pouvez pas supprimer une politique de masquage qui est toujours attachée à une ou plusieurs tables. Pour plus d'informations sur le masquage dynamique des données, consultez Masquage dynamique des données.

Les super-utilisateurs et les utilisateurs ou les rôles disposant du rôle sys:secadmin peuvent supprimer une politique de masquage.

Syntaxe

```
DROP MASKING POLICY policy_name;
```

Paramètres

policy_name

Nom de la politique de masquage à supprimer.

DROP MODEL

Supprime un modèle de la base de données. Seul le propriétaire du modèle ou un super-utilisateur peut supprimer un modèle.

DROP MODEL supprime également toutes les fonctions de prédiction associées dérivées de ce modèle, tous les artefacts Amazon Redshift liés au modèle et toutes les données Amazon S3 liées au modèle. Pendant que le modèle est toujours en cours de formation sur Amazon SageMaker AI, DROP MODEL annulera ces opérations.

La commande n’est pas réversible. La commande DROP MODEL valide immédiatement.

Autorisations requises

Les autorisations suivantes sont requises pour DROP MODEL :

- Superuser
- Utilisateurs disposant de l’autorisation DROP MODEL
- Propriétaire du modèle
- Propriétaire du schéma

Syntaxe

```
DROP MODEL [ IF EXISTS ] model_name
```

Paramètres

IF EXISTS

Clause indiquant que si le schéma spécifié existe déjà, la commande ne doit apporter aucune modification et renvoyer un message selon lequel le schéma existe.

model_name

Nom du modèle. Le nom du modèle d'un schéma doit être unique.

Exemples

L'exemple suivant supprime le modèle demo_ml.customer_churn.

```
DROP MODEL demo_ml.customer_churn
```

DROP MATERIALIZED VIEW

Supprime une vue matérialisée.

Pour plus d'informations sur les vues matérialisées, consultez Vues matérialisées dans Amazon Redshift.

Syntaxe

```
DROP MATERIALIZED VIEW [ IF EXISTS ] mv_name [ CASCADE | RESTRICT ]
```

Paramètres

IF EXISTS

Clause qui spécifie de vérifier si la vue matérialisée nommée existe. Si la vue matérialisée n'existe pas, la commande `DROP MATERIALIZED VIEW` renvoie un message d'erreur. Cette clause est utile lors de la rédaction des scripts, afin d'éviter l'échec de ces derniers si vous supprimez une vue matérialisée inexistante.

mv_name

Nom de la vue matérialisée à supprimer.

CASCADE

Clause indiquant de supprimer automatiquement les objets dont dépend la vue matérialisée, tels que les autres vues.

RESTRICT

Une clause qui indique de ne pas supprimer la vue matérialisée si des objets en dépendent. Il s'agit de l'option par défaut.

Notes d'utilisation

Seul le propriétaire d'une vue matérialisée peut utiliser `DROP MATERIALIZED VIEW` sur cette vue. Un superutilisateur ou un utilisateur auquel des privilèges DROP ont été spécifiquement accordés peuvent faire exception à cette règle.

Lorsque vous écrivez une instruction DROP pour une vue matérialisée et qu'il existe une vue de même nom, vous obtenez une erreur qui vous demande d'utiliser DROP VIEW. Une erreur se produit même dans le cas où vous utilisez `DROP MATERIALIZED VIEW IF EXISTS`.

exemple

L'exemple suivant supprime la vue matérialisée `tickets_mv`.

```
DROP MATERIALIZED VIEW tickets_mv;
```

DROP PROCEDURE

Supprime une procédure. Pour supprimer une procédure, le nom de la procédure et les types de données d'argument en entrée (signature), sont requis. Si vous le souhaitez, vous pouvez inclure l'ensemble des types de données d'argument, y compris les arguments OUT. Pour trouver la signature d'une procédure, utilisez la commande SHOW PROCEDURE. Pour plus d'informations sur les signatures d'une procédure, consultez PG_PROC_INFO.

Privilèges requis

Les privilèges suivants sont requis pour DROP PROCEDURE :

- Superuser
- Utilisateurs disposant du privilège DROP PROCEDURE

- Propriétaire de la procédure

Syntaxe

```
DROP PROCEDURE sp_name ( [ [ argname ] [ argmode ] argtype [, ...] ] )
```

Paramètres

sp_name

Nom de la procédure à supprimer.

argname

Nom d'un argument d'entrée. DROP PROCEDURE ignore les noms d'argument, car seuls les types de données des arguments sont nécessaires pour déterminer l'identité de la procédure.

argmode

Mode d'un argument, à savoir IN, OUT ou INOUT. Les arguments OUT ne sont pas obligatoires, car ils ne servent pas à identifier une procédure stockée.

argtype

Type de données de l'argument en entrée. Pour afficher la liste des types de données pris en charge, consultez Types de données.

Exemples

L'exemple suivant supprime une procédure stockée appelée `quarterly_revenue`.

```
DROP PROCEDURE quarterly_revenue(volume INOUT bigint, at_price IN numeric,result OUT
 int);
```

DROP RLS POLICY

Supprime une politique de sécurité de niveau des lignes pour toutes les tables de toutes les bases de données.

Les super-utilisateurs, les utilisateurs ou les utilisateurs disposant du rôle sys:secadmin peuvent supprimer une stratégie.

Syntaxe

```
DROP RLS POLICY [ IF EXISTS ] policy_name [ CASCADE | RESTRICT ]
```

Paramètres

IF EXISTS

Clause qui indique si la politique spécifiée existe déjà.

policy_name

Nom de la politique .

CASCADE

Clause qui indique de détacher automatiquement la politique de toutes les tables attachées avant de la supprimer.

RESTRICT

Clause qui indique de ne pas supprimer la politique lorsqu'elle est attachée à certaines tables. Il s'agit de l'option par défaut.

Exemples

L'exemple suivant supprime la politique de sécurité de niveau des lignes.

```
DROP RLS POLICY policy_concerts;
```

DROP ROLE

Supprime un rôle d'une base de données. Seul le propriétaire du rôle qui a créé le rôle, un utilisateur avec l'option WITH ADMIN ou un super-utilisateur peut supprimer un rôle.

Vous ne pouvez pas supprimer un rôle accordé à un utilisateur ou à un autre rôle dépendant de ce rôle.

Privilèges requis

Voici les privilèges requis pour DROP ROLE :

- Superuser
- Propriétaire du rôle qui est soit l'utilisateur qui a créé le rôle, soit un utilisateur qui s'est vu accorder le rôle avec le privilège WITH ADMIN OPTION.

Syntaxe

```
DROP ROLE role_name [ FORCE | RESTRICT ]
```

Paramètres

role_name

Nom du rôle.

[FORCE | RESTRICT]

Le paramètre par défaut est RESTRICT. Amazon Redshift génère une erreur lorsque vous tentez de supprimer un rôle qui a hérité d'un autre rôle. Utilisez FORCE pour supprimer, le cas échéant, toutes les affectations de rôle.

Exemples

L'exemple suivant supprime le rôle `sample_role`.

```
DROP ROLE sample_role FORCE;
```

L'exemple suivant tente de supprimer le rôle sample_role1 accordé à un utilisateur avec l'option RESTRICT par défaut.

```
CREATE ROLE sample_role1;
GRANT ROLE sample_role1 TO user1;
DROP ROLE sample_role1;
ERROR:  cannot drop this role since it has been granted on a user
```

Pour supprimer correctement le rôle sample_role1 qui a été accordé à un utilisateur, utilisez l'option FORCE.

```
DROP ROLE sample_role1 FORCE;
```

L'exemple suivant tente de supprimer le rôle sample_role2 dont un autre rôle dépend avec l'option RESTRICT par défaut.

```
CREATE ROLE sample_role1;
CREATE ROLE sample_role2;
GRANT ROLE sample_role1 TO sample_role2;
DROP ROLE sample_role2;
ERROR:  cannot drop this role since it depends on another role
```

Pour supprimer correctement le rôle sample_role2 dont un autre rôle dépend, utilisez l'option FORCE.

```
DROP ROLE sample_role2 FORCE;
```

DROP SCHEMA

Supprime un schéma. Pour un schéma externe, vous pouvez également supprimer la base de données externe associée au schéma. La commande n'est pas réversible.

Privilèges requis

Les privilèges suivants sont requis pour DROP SCHEMA :

- Superuser
- Propriétaire du schéma
- Utilisateurs disposant du privilège DROP SCHEMA

Syntaxe

```
DROP SCHEMA [ IF EXISTS ] name [, ...]
[ DROP EXTERNAL DATABASE ]
[ CASCADE | RESTRICT ]
```

Paramètres

IF EXISTS

Clause indiquant que si le schéma spécifié n'existe pas, la commande ne doit faire aucune modification et renvoyer un message selon lequel le schéma n'existe pas, plutôt que de mettre fin avec une erreur.

Comme cette clause est utile lors de l'écriture de scripts, le script n'échoue pas si DROP SCHEMA s'exécute sur un schéma qui n'existe pas.

nom

Noms des schémas à supprimer. Vous pouvez spécifier plusieurs noms de schémas séparés par des virgules.

DROP EXTERNAL DATABASE

Clause qui indique que si un schéma externe est supprimé, il convient de supprimer la base de données externe associée au schéma externe, si elle existe. Si aucune base de données externe n'existe, la commande renvoie un message indiquant qu'aucune base de données externe n'existe. Si plusieurs schémas externes sont supprimés, toutes les bases de données associées aux schémas spécifiés sont supprimées.

Si une base de données externe contient des objets dépendants tels que des tables, incluez l'option CASCADE pour supprimer également ces objets dépendants.

Lorsque vous supprimez une base de données externe, cette base de données est également supprimée pour tous les autres schémas externes associés à la base de données. Les tables définies dans d'autres schémas externes utilisant cette base de données sont également supprimées.

DROP EXTERNAL DATABASE ne prend pas en charge les bases de données externes stockées dans un metastore HIVE.

CASCADE

Mot-clé qui indique de supprimer automatiquement tous les objets figurant dans le schéma. Si DROP EXTERNAL DATABASE est spécifiée, tous les objets figurant dans la base de données externe sont également supprimés.

RESTRICT

Mot-clé qui indique de ne pas supprimer un schéma ou une base de données externe s'il contient ou si elle contient des objets. Il s'agit de l'action par défaut.

exemple

L'exemple suivant supprime un schéma nommé S_SALES. Cet exemple utilise RESTRICT comme mécanisme de sécurité de telle sorte que le schéma n'est pas supprimé s'il contient des objets. Dans ce cas, vous devez supprimer les objets du schéma avant de supprimer le schéma.

```
drop schema s_sales restrict;
```

L'exemple suivant supprime un schéma nommé S_SALES et tous les objets qui dépendent de ce schéma.

```
drop schema s_sales cascade;
```

L'exemple suivant supprime le schéma S_SALES s'il existe, ou ne fait rien et renvoie un message dans le cas contraire.

```
drop schema if exists s_sales;
```

L'exemple suivant supprime un schéma externe nommé S_SPECTRUM et la base de données externe qui lui est associée. Cet exemple utilise RESTRICT pour ne pas supprimer le schéma et la base de données s'ils contiennent des objets. Dans ce cas, vous devez supprimer les objets dépendants avant de supprimer le schéma et la base de données.

```
drop schema s_spectrum drop external database restrict;
```

L'exemple suivant supprime plusieurs schémas et les bases de données externes qui leur sont associées, ainsi que tous les objets dépendants éventuels.

```
drop schema s_sales, s_profit, s_revenue drop external database cascade;
```

DROP TABLE

Supprime une table d'une base de données.

Si vous essayez de vider une table de lignes, sans supprimer la table, utilisez la commande DELETE ou TRUNCATE.

DROP TABLE supprime les contraintes qui existent sur la table cible. Plusieurs tables peuvent être supprimées avec une seule commande DROP TABLE.

Vous ne pouvez pas exécuter DROP TABLE avec une table externe à l'intérieur d'une transaction (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

Pour trouver un exemple dans lequel le privilège DROP est accordé à un groupe, consultez GRANT Exemples.

Privilèges requis

Les privilèges suivants sont requis pour DROP TABLE :

- Superuser
- Utilisateurs disposant du privilège DROP TABLE
- Propriétaire de table disposant du privilège USAGE sur le schéma

Syntaxe

```
DROP TABLE [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT ]
```

Paramètres

IF EXISTS

Clause indiquant que si la table spécifiée n'existe pas, la commande ne doit faire aucune modification et renvoyer un message selon lequel la table n'existe pas, plutôt que de mettre fin avec une erreur.

Comme cette clause est utile lors de l'écriture de scripts, le script n'échoue pas si DROP TABLE s'exécute sur une table qui n'existe pas.

nom

Nom de la table à supprimer.

CASCADE

Clause qui indique de supprimer automatiquement les objets qui dépendent de la table, tels que les vues.

Pour créer une vue qui ne dépend pas d'autres objets de base de données, tels que des vues et des tables, incluez la clause WITH NO SCHEMA BINDING dans la définition de vue. Pour plus d'informations, consultez CREATE VIEW.

RESTRICT

Clause qui indique de ne pas pour supprimer la table si des objets en dépendent. Il s'agit de l'action par défaut.

Exemples

Suppression d’une table sans dépendances

L’exemple suivant crée et supprime une table appelée FEEDBACK qui n’a pas de dépendances :

```
create table feedback(a int);

drop table feedback;
```

Si une table contient des colonnes qui sont référencées par des vues ou par d’autres tables, Amazon Redshift affiche un message tel que le suivant.

```
Invalid operation: cannot drop table feedback because other objects depend on it
```

Suppression simultanée de deux tables

L’ensemble de commandes suivant crée une table FEEDBACK et une table BUYERS, puis supprime les deux tables avec une seule commande :

```
create table feedback(a int);

create table buyers(a int);

drop table feedback, buyers;
```

Suppression d’une table avec une dépendance

Les étapes suivantes montrent comment supprimer une table appelée FEEDBACK à l’aide de l’option CASCADE.

Commencez par créer une simple table appelée FEEDBACK en utilisant la commande CREATE TABLE :

```
create table feedback(a int);
```

Puis, utilisez la commande CREATE VIEW pour créer une vue appelée FEEDBACK_VIEW reposant sur la table FEEDBACK :

```
create view feedback_view as select * from feedback;
```

L'exemple suivant supprime la table FEEDBACK, ainsi que la vue FEEDBACK_VIEW, car FEEDBACK_VIEW dépend de la table FEEDBACK :

```
drop table feedback cascade;
```

Affichage des dépendances pour une table

Pour renvoyer les dépendances de votre table, utilisez l'exemple suivant. Remplacez *my_schema* et *my_table* par votre propre schéma et table.

```
SELECT dependent_ns.nspname as dependent_schema
, dependent_view.relname as dependent_view
, source_ns.nspname as source_schema
, source_table.relname as source_table
, pg_attribute.attname as column_name
FROM pg_depend
JOIN pg_rewrite ON pg_depend.objid = pg_rewrite.oid
JOIN pg_class as dependent_view ON pg_rewrite.ev_class = dependent_view.oid
JOIN pg_class as source_table ON pg_depend.refobjid = source_table.oid
JOIN pg_attribute ON pg_depend.refobjid = pg_attribute.attrelid
    AND pg_depend.refobjsubid = pg_attribute.attnum
JOIN pg_namespace dependent_ns ON dependent_ns.oid = dependent_view.relnamespace
JOIN pg_namespace source_ns ON source_ns.oid = source_table.relnamespace
WHERE
source_ns.nspname = 'my_schema'
AND source_table.relname = 'my_table'
AND pg_attribute.attnum > 0
ORDER BY 1,2
LIMIT 10;
```

Pour supprimer *my_table* et ses dépendances, utilisez l'exemple suivant. Cet exemple renvoie également toutes les dépendances de la table supprimée.

```
DROP TABLE my_table CASCADE;

SELECT dependent_ns.nspname as dependent_schema
, dependent_view.relname as dependent_view
, source_ns.nspname as source_schema
, source_table.relname as source_table
, pg_attribute.attname as column_name
FROM pg_depend
JOIN pg_rewrite ON pg_depend.objid = pg_rewrite.oid
```

```
JOIN pg_class as dependent_view ON pg_rewrite.ev_class = dependent_view.oid
JOIN pg_class as source_table ON pg_depend.refobjid = source_table.oid
JOIN pg_attribute ON pg_depend.refobjid = pg_attribute.attrelid
    AND pg_depend.refobjsubid = pg_attribute.attnum
JOIN pg_namespace dependent_ns ON dependent_ns.oid = dependent_view.relnamespace
JOIN pg_namespace source_ns ON source_ns.oid = source_table.relnamespace
WHERE
source_ns.nspname = 'my_schema'
AND source_table.relname = 'my_table'
AND pg_attribute.attnum > 0
ORDER BY 1,2
LIMIT 10;

+-----------------+---------------+--------------+-------------+-------------+
| dependent_schema | dependent_view | source_schema | source_table | column_name |
+-----------------+---------------+--------------+-------------+-------------+
```

Suppression d’une table avec IF EXISTS

L’exemple suivant supprime la table FEEDBACK si elle existe, ou ne fait rien et renvoie un message dans le cas contraire :

```
drop table if exists feedback;
```

DROP USER

Supprime un utilisateur d’une base de données. Plusieurs utilisateurs peuvent être supprimés avec une seule commande DROP USER. Vous devez être un superutilisateur de base de données ou disposer de l'autorisation DROP USER pour exécuter cette commande.

Syntaxe

```
DROP USER [ IF EXISTS ] name [, ... ]
```

Paramètres

IF EXISTS

Clause indiquant que si l’utilisateur spécifié n’existe pas, la commande ne doit faire aucune modification et renvoyer un message selon lequel l’utilisateur n’existe pas, plutôt que de mettre fin avec une erreur.

Comme cette clause est utile lors de l'écriture de scripts, le script n'échoue pas si DROP USER s'exécute sur un utilisateur non existant.

nom

Nom de l'utilisateur à supprimer. Vous pouvez spécifier plusieurs utilisateurs, avec une virgule séparant chaque utilisateur du suivant.

Notes d'utilisation

Vous ne pouvez pas supprimer l'utilisateur nommé `rdsdb` ou l'utilisateur administrateur de la base de données qui est généralement nommée `awsuser` ou `admin`.

Vous ne pouvez pas supprimer un utilisateur si l'utilisateur possède un objet de base de données, tel qu'un schéma, une base de données, une table ou une vue, ou si l'utilisateur a des privilèges sur une base de données, une table, une colonne ou un groupe. Si vous tentez de supprimer un tel utilisateur, vous recevez l'une des erreurs suivantes.

```
ERROR: user "username" can't be dropped because the user owns some object [SQL
 State=55006]

ERROR: user "username" can't be dropped because the user has a privilege on some object
 [SQL State=55006]
```

Pour obtenir des instructions détaillées sur la façon de trouver les objets appartenant à un utilisateur de base de données, consultez Comment résoudre l'erreur « l'utilisateur ne peut pas être supprimé » dans Amazon Redshift ? dans le Centre de connaissances.

> **Note**
>
> Amazon Redshift vérifie seulement la base de données actuelle avant de supprimer un utilisateur. DROP USER ne renvoie une erreur que si l'utilisateur possède des objets de base de données ou a des privilèges sur des objets d'une autre base de données. Si vous supprimez un utilisateur qui possède des objets d'une autre base de données, le propriétaire de ces objets acquiert l'état « unknown ».

Si un utilisateur possède un objet, supprimez d'abord l'objet ou remplacez sa propriété par celle d'un autre utilisateur avant de supprimer l'utilisateur original. Si l'utilisateur dispose de privilèges pour

un objet, révoquez d'abord les privilèges avant de supprimer l'utilisateur. L'exemple suivant montre la suppression d'un objet, la modification de la propriété et la révocation des privilèges avant de supprimer l'utilisateur.

```
drop database dwdatabase;
alter schema dw owner to dwadmin;
revoke all on table dwtable from dwuser;
drop user dwuser;
```

Exemples

L'exemple suivant supprime un utilisateur appelé paulo :

```
drop user paulo;
```

L'exemple suivant supprime deux utilisateurs, paulo et martha :

```
drop user paulo, martha;
```

L'exemple suivant supprime l'utilisateur paulo s'il existe, ou ne fait rien et renvoie un message dans le cas contraire :

```
drop user if exists paulo;
```

DROP VIEW

Supprime une vue de la base de données. Plusieurs vues peuvent être supprimées avec une seule commande DROP VIEW. La commande n'est pas réversible.

Privilèges requis

Les privilèges suivants sont requis pour DROP VIEW :

- Superuser
- Utilisateurs disposant du privilège DROP VIEW
- Propriétaire de la vue

Syntaxe

```
DROP VIEW [ IF EXISTS ] name [, ... ] [ CASCADE | RESTRICT ]
```

Paramètres

IF EXISTS

Clause indiquant que si la vue spécifiée n'existe pas, la commande ne doit faire aucune modification et renvoyer un message selon lequel la vue n'existe pas, plutôt que de mettre fin avec une erreur.

Comme cette clause est utile lors de l'écriture de scripts, le script n'échoue pas si DROP VIEW s'exécute sur une vue non existante.

nom

Nom de la vue à supprimer.

CASCADE

Clause qui indique de supprimer automatiquement les objets qui dépendent de la vue, tels que d'autres vues.

Pour créer une vue qui ne dépend pas d'autres objets de base de données, tels que des vues et des tables, incluez la clause WITH NO SCHEMA BINDING dans la définition de vue. Pour de plus amples informations, veuillez consulter CREATE VIEW.

Notez que si vous incluez CASCADE et que le nombre d'objets de base de données supprimés atteint dix ou plus, il est possible que votre client de base de données ne répertorie pas tous les objets supprimés dans les résultats récapitulatifs. Cela est généralement dû au fait que les outils client SQL ont des limites par défaut sur les résultats renvoyés.

RESTRICT

Clause qui indique de ne pas supprimer la vue si des objets en dépendent. Il s'agit de l'action par défaut.

Exemples

L'exemple suivant supprime la vue appelée event :

```
drop view event;
```

Pour supprimer une vue qui comporte des dépendances, utilisez l'option CASCADE. Par exemple, imaginons que nous commencions avec une table appelée EVENT. Nous créons ensuite la vue eventview de la table EVENT, à l'aide de la commande CREATE VIEW, comme illustré dans l'exemple suivant :

```
create view eventview as
select dateid, eventname, catid
from event where catid = 1;
```

Maintenant, nous créons une deuxième vue appelée myeventview, qui est basée sur la première vue eventview :

```
create view myeventview as
select eventname, catid
from eventview where eventname <> ' ';
```

À ce stade, deux vues ont été créées : eventview et myeventview.

La vue myeventview est une vue enfant avec eventview en tant que parent.

Pour supprimer la vue eventview, la commande évidente à utiliser est la suivante :

```
drop view eventview;
```

Notez que si vous exécutez cette commande dans ce cas, vous obtenez l'erreur suivante :

```
drop view eventview;
ERROR: can't drop view eventview because other objects depend on it
HINT: Use DROP ... CASCADE to drop the dependent objects too.
```

Pour résoudre le problème, exécutez la commande suivante (comme indiqué dans le message d'erreur) :

```
drop view eventview cascade;
```

Les deux vues eventview et myeventview ont été supprimées avec succès.

L'exemple suivant supprime la vue eventview si elle existe, ou ne fait rien et renvoie un message dans le cas contraire :

```
drop view if exists eventview;
```

FIN

Valide la transaction actuelle. Effectue exactement la même fonction que la commande COMMIT.

Consultez COMMIT pour une documentation plus détaillée.

Syntaxe

```
END [ WORK | TRANSACTION ]
```

Paramètres

WORK

Mot-clé facultatif.

TRANSACTION

Mot-clé facultatif ; WORK et TRANSACTION sont synonymes.

Exemples

Les exemples suivants mettent tous fin au bloc de transaction et valident la transaction :

```
end;
```

```
end work;
```

```
end transaction;
```

Après une de ces commandes, Amazon Redshift termine le bloc de transaction et valide les modifications.

EXECUTE

Exécute une instruction préalablement préparée.

Syntaxe

```
EXECUTE plan_name [ (parameter [, ...]) ]
```

Paramètres

nom_plan

Nom de l'instruction préparée à exécuter.

paramètre

Valeur réelle d'un paramètre de l'instruction préparée. Ce doit être une expression générant une valeur dont le type est compatible avec le type de données spécifié pour cette position de paramètre de la commande PREPARE qui a créé l'instruction préparée.

Notes d'utilisation

EXECUTE permet d'exécuter une requête préalablement préparée. Comme les instructions préparées existent uniquement pendant la durée d'une séance, l'instruction préparée doit avoir été créée par une instruction PREPARE exécutée plus tôt dans la séance en cours.

Si l'instruction PREPARE précédente spécifiait certains paramètres, un ensemble compatible de paramètres doit être passé à l'instruction EXECUTE ou Amazon Redshift renvoie une erreur. Contrairement aux fonctions, les instructions préparées ne sont pas surchargées en fonction du type ou du nombre de paramètres spécifiés ; le nom d'une instruction préparée doit être unique au sein d'une séance de base de données.

Quand une commande EXECUTE est émise pour l'instruction préparée, Amazon Redshift peut le cas échéant réviser le plan d'exécution de la requête (pour améliorer les performances en fonction des valeurs de paramètre spécifiées) avant l'exécution de l'instruction préparée. En outre, pour chaque nouvelle exécution d'une instruction préparée, Amazon Redshift peut réviser le plan d'exécution de la requête en fonction des différentes valeurs de paramètre spécifiées avec l'instruction EXECUTE. Pour examiner le plan d'exécution de la requête qu'Amazon Redshift a choisi pour les instructions EXECUTE données, utilisez la commande EXPLAIN.

Pour obtenir des exemples et plus d'informations sur la création et l'utilisation des instructions préparées, consultez PREPARE.

Consultez aussi

DEALLOCATE, PREPARE

EXPLAIN

Affiche le plan d'exécution d'une instruction de requête sans exécution de la requête. Pour plus d'informations sur le flux de travail d'analyse des requêtes, consultez Flux de travail d'analyse des requêtes.

Syntaxe

```
EXPLAIN [ VERBOSE ] query
```

Paramètres

VERBOSE

Affiche le plan de requête complet au lieu d'un simple résumé.

query

Instruction de la requête à expliquer. La requête peut être une instruction SELECT, INSERT, CREATE TABLE AS, UPDATE ou DELETE.

Notes d'utilisation

Les performances d'EXPLAIN sont parfois influencées par le temps nécessaire à la création de tables temporaires. Par exemple, une requête qui utilise l'optimisation courante des sous-expressions nécessite la création et l'analyse de tables temporaires pour renvoyer la sortie EXPLAIN. Le plan de requête dépend du schéma et des statistiques des tables temporaires. Par conséquent, la commande EXPLAIN pour ce type de requête peut prendre plus de temps pour s'exécuter que prévu.

Vous pouvez utiliser EXPLAIN uniquement pour les commandes suivantes :

- SELECT

- SELECT INTO
- CREATE TABLE AS
- INSERT
- UPDATE
- DELETE

La commande EXPLAIN échoue si vous l'utilisez pour d'autres commandes SQL, telles que le langage de définition de données (DDL) ou les opérations de base de données.

Les coûts unitaires relatifs de sortie EXPLAIN sont utilisés par Amazon Redshift pour choisir un plan de requête. Amazon Redshift compare les tailles de différentes estimations de ressources pour déterminer le plan.

Étapes de planification et d'exécution de la requête

Le plan d'exécution d'une instruction de requête Amazon Redshift spécifique ventile l'exécution et le calcul d'une requête en une séquence discrète d'étapes et d'opérations de table qui produisent un ensemble de résultats final pour la requête. Pour obtenir des informations sur la planification des requêtes, consultez Traitement des requêtes.

Le tableau suivant fournit un résumé des étapes qu'Amazon Redshift peut utiliser dans le développement d'un plan d'exécution dans le cas d'une requête qu'un utilisateur soumet pour l'exécution.

Opérateurs EXPLAIN	Étapes de l'exécution d'une requête	Description
SCAN :		
Analyse séquentielle	scan	Opérateur ou étape d'analyse de table ou d'analyse de relation Amazon Redshift. Analyse toute la table de manière séquentielle du début à la fin ; analyse aussi les contraintes de requête pour chaque ligne (Filter) si la clause WHERE est spécifiée. Permet aussi d'exécuter les instructions INSERT, UPDATE et DELETE.

Opérateurs EXPLAIN	Étapes de l'exécution d'une requête	Description
JOINS : Amazon Redshift utilise différents opérateurs de jointure basés sur la conception physique des tables jointes, l'emplacement des données requises pour la jointure et les attributs spécifiques de la requête elle-même. Analyse de sous-requête : l'analyse et l'ajout de sous-requête permettent d'exécuter les requêtes UNION.		
Boucle imbriquée	nloop	Jointure la moins optimale ; principalement utilisée pour les jointures croisées (produits cartésiens, sans condition de jointure) et certaines jointures d'inégalité.
Joindre par hachage	hjoin	Également utilisé pour les jointures internes et les jointures externes gauche et droite, et généralement plus rapide qu'une jointure de boucle imbriquée. La jointure de hachage lit la table externe, hache la colonne de jointure et recherche les correspondances de la table de hachage interne. L'étape peut déverser sur le disque. (L'entrée interne de hjoin est une étape de hachage qui peut être basée sur le disque.)
Joindre par fusion	mjoin	Également utilisé pour les jointures internes et les jointures externes (pour les tables de jointure qui sont distribuées et triées sur les colonnes de jointure). Généralement, l'algorithme de jointure Amazon Redshift le plus rapide, sans inclure d'autres considérations de coût.
AGGREGATION : opérateurs et étapes utilisés pour les requêtes impliquant les fonctions d'agrégation et les opérations GROUP BY.		
Regrouper	aggr	Opérateur/étape pour les fonctions d'agrégation scalaires.

Opérateurs EXPLAIN	Étapes de l'exécution d'une requête	Description
HashAggregate	aggr	Opérateur/étape pour les fonctions d'agrégation groupées. Peut fonctionner à partir du disque en raison du déversement de la table de hachage sur le disque.
GroupAggregate	aggr	Opérateur parfois choisi pour les requêtes agrégation groupées si le paramètre de configuration Amazon Redshift pour le paramètre force_hash_grouping est désactivé.
SORT : opérateurs et étapes utilisés lorsque les requêtes doivent trier ou fusionner des jeux de résultats.		
Tri	sort	Sort effectue le tri spécifié par la clause ORDER BY ainsi que d'autres opérations telles que UNIONs les jointures. Peut fonctionner à partir du disque.
Fusionner	merge	Produit les résultats finaux triés d'une requête basée sur les résultats intermédiaires triés provenant d'opérations effectuées en parallèle.
Opérations EXCEPT, INTERSECT et UNION :		
SetOp Sauf [Distinct]	hjoin	Utilisé pour les requêtes EXCEPT. Peut fonctionner à partir du disque en fonction du fait que l'entrée de hachage peut être basée sur le disque.
Hash Intersect [Distinct]	hjoin	Utilisé pour les requêtes INTERSECT. Peut fonctionner à partir du disque en fonction du fait que l'entrée de hachage peut être basée sur le disque.

Opérateurs EXPLAIN	Étapes de l'exécution d'une requête	Description
Append [All |Distinct]	enregistrer	Utilisé avec l'analyse de sous-requête pour implémenter les requêtes UNION et UNION ALL. Peut fonctionner à partir du disque grâce à « save ».
Divers/autres :		
Hachage	hachage	Utilisé pour les jointures internes et les jointures externes gauche et droite (fournit une entrée à une jointure de hachage). L'opérateur de hachage crée la table de hachage pour la table interne d'une jointure. (La table interne est celle dans laquelle les correspondances sont vérifiées et, dans une jointure de deux tables, elle est généralement la plus petite des deux.)
Limite	limite	Évalue la clause LIMIT.
Materialize	enregistrer	Matérialise les lignes pour l'entrée des jointures de boucles imbriquées et quelques autres jointures de fusion. Peut fonctionner à partir du disque.
--	parse	Utilisé pour analyser les données d'entrée texte pendant un chargement.
--	project	Permet de réorganiser les colonnes et les expressions de calcul, à savoir les données du projet.
Résultat	--	Exécute les fonctions scalaires qui n'impliquent pas un accès aux tables.
--	return	renvoie les lignes au principal ou au client.

Opérateurs EXPLAIN	Étapes de l'exécution d'une requête	Description
Subplan	--	Utilisé pour certaines sous-requêtes.
Unique	unique	Supprime les doublons des requêtes SELECT DISTINCT et UNION.
Fenêtre	window	Fonctions d'agrégation de calcul et de fenêtrage de classement. Peut fonctionner à partir du disque.
Opérations de réseau :		
Network (Broadcast)	bcast	Broadcast est également un attribut des opérateurs et des étapes Join Explain.
Network (Distribute)	dist	Distribue les lignes sur les nœuds de calcul pour le traitement parallèle par cluster d'entrepôt des données.
Network (Send to Leader)	return	Renvoie les résultats vers le principal en vue d'un traitement ultérieur.
Opérations DML (opérateurs qui modifient les données) :		
Insert (using Result)	insert	Insère les données.
Delete (Scan + Filter)	supprimer	Supprime les données. Peut fonctionner à partir du disque.
Update (Scan + Filter)	delete, insert	Implémenté comme Delete et Insert.

Utilisation d'EXPLAIN pour RLS

Si une requête contient une table soumise à des politiques de sécurité au niveau des lignes (RLS), EXPLAIN affiche un nœud RLS spécial. SecureScan Amazon Redshift enregistre également le même type de nœud dans la table système STL_EXPLAIN. EXPLAIN ne révèle pas le prédicat RLS qui

s’applique à dim_tbl. Le type de SecureScan nœud RLS indique que le plan d'exécution contient des opérations supplémentaires invisibles pour l'utilisateur actuel.

L'exemple suivant illustre un SecureScan nœud RLS.

```
EXPLAIN
SELECT D.cint
FROM fact_tbl F INNER JOIN dim_tbl D ON F.k_dim = D.k
WHERE F.k_dim / 10 > 0;
                              QUERY PLAN
----------------------------------------------------------------------
 XN Hash Join DS_DIST_ALL_NONE  (cost=0.08..0.25 rows=1 width=4)
   Hash Cond: ("outer".k_dim = "inner"."k")
   ->  *XN* *RLS SecureScan f  (cost=0.00..0.14 rows=2 width=4)*
         Filter: ((k_dim / 10) > 0)
   ->  XN Hash  (cost=0.07..0.07 rows=2 width=8)
         ->  XN Seq Scan on dim_tbl d  (cost=0.00..0.07 rows=2 width=8)
               Filter: (("k" / 10) > 0)
```

Pour permettre une enquête complète des plans de requêtes soumis au RLS, Amazon Redshift propose les autorisations système EXPLAIN RLS. Les utilisateurs auxquels cette autorisation a été accordée peuvent inspecter des plans de requêtes complets qui comprennent également des prédicats RLS.

L'exemple suivant illustre un Seq Scan supplémentaire situé sous le SecureScan nœud RLS et inclut également le prédicat de politique RLS (k_dim > 1).

```
EXPLAIN SELECT D.cint
FROM fact_tbl F INNER JOIN dim_tbl D ON F.k_dim = D.k
WHERE F.k_dim / 10 > 0;
                                  QUERY PLAN
------------------------------------------------------------------------------
 XN Hash Join DS_DIST_ALL_NONE  (cost=0.08..0.25 rows=1 width=4)
   Hash Cond: ("outer".k_dim = "inner"."k")
   *->  XN RLS SecureScan f  (cost=0.00..0.14 rows=2 width=4)
         Filter: ((k_dim / 10) > 0)*
         ->  *XN* *Seq Scan on fact_tbl rls_table  (cost=0.00..0.06 rows=5 width=8)
               Filter: (k_dim > 1)*
   ->  XN Hash  (cost=0.07..0.07 rows=2 width=8)
         ->  XN Seq Scan on dim_tbl d  (cost=0.00..0.07 rows=2 width=8)
               Filter: (("k" / 10) > 0)
```

Lorsque l'autorisation EXPLAIN RLS est accordée à un utilisateur, Amazon Redshift enregistre le plan de requêtes complet, y compris les prédicats RLS dans la table système STL_EXPLAIN. Les requêtes qui sont exécutées alors que cette autorisation n'est pas accordée seront enregistrées sans les données internes RLS. Accorder ou supprimer l'autorisation EXPLAIN RLS ne changera pas ce qu'Amazon Redshift a enregistré dans STL_EXPLAIN pour les requêtes précédentes.

Relations Redshift protégées par AWS Lake Formation-RLS

L'exemple suivant illustre un SecureScan nœud LF, que vous pouvez utiliser pour visualiser les relations Lake Formation-RLS.

```
EXPLAIN
SELECT *
FROM lf_db.public.t_share
WHERE a > 1;
QUERY PLAN
------------------------------------------------------------
XN LF SecureScan t_share  (cost=0.00..0.02 rows=2 width=11)
(2 rows)
```

Exemples

> **Note**
>
> Pour ces exemples, l'exemple de sortie peut varier selon la configuration d'Amazon Redshift.

L'exemple suivant renvoie le plan de requête pour une requête qui sélectionne EVENTID, EVENTNAME, VENUEID et VENUENAME à partir des tables EVENT et VENUE :

```
explain
select eventid, eventname, event.venueid, venuename
from event, venue
where event.venueid = venue.venueid;
```

```
                              QUERY PLAN
------------------------------------------------------------------------
XN Hash Join DS_DIST_OUTER  (cost=2.52..58653620.93 rows=8712 width=43)
Hash Cond: ("outer".venueid = "inner".venueid)
->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=23)
```

```
->  XN Hash  (cost=2.02..2.02 rows=202 width=22)
->  XN Seq Scan on venue  (cost=0.00..2.02 rows=202 width=22)
(5 rows)
```

L'exemple suivant renvoie le plan de requête pour la même requête avec la sortie des commentaires :

```
explain verbose
select eventid, eventname, event.venueid, venuename
from event, venue
where event.venueid = venue.venueid;
```

```
                                QUERY PLAN
------------------------------------------------------------------------------
{HASHJOIN
:startup_cost 2.52
:total_cost 58653620.93
:plan_rows 8712
:plan_width 43
:best_pathkeys <>
:dist_info DS_DIST_OUTER
:dist_info.dist_keys (
TARGETENTRY
{
VAR
:varno 2
:varattno 1
...

XN Hash Join DS_DIST_OUTER  (cost=2.52..58653620.93 rows=8712 width=43)
Hash Cond: ("outer".venueid = "inner".venueid)
->  XN Seq Scan on event  (cost=0.00..87.98 rows=8798 width=23)
->  XN Hash  (cost=2.02..2.02 rows=202 width=22)
->  XN Seq Scan on venue  (cost=0.00..2.02 rows=202 width=22)
(519 rows)
```

L'exemple suivant renvoie le plan de requête pour une instruction CREATE TABLE AS (CTAS) :

```
explain create table venue_nonulls as
select * from venue
where venueseats is not null;
```

```
QUERY PLAN
-------------------------------------------------------------
XN Seq Scan on venue  (cost=0.00..2.02 rows=187 width=45)
Filter: (venueseats IS NOT NULL)
(2 rows)
```

FETCH

Extrait les lignes à l'aide d'un curseur. Pour obtenir des informations sur la déclaration d'un curseur, consultez DECLARE.

FETCH extrait les lignes en fonction de la position actuelle du curseur. Lorsqu'un curseur est créé, il est placé avant la première ligne. Après une opération FETCH, le curseur est placé sur la dernière ligne récupérée. Si FETCH s'exécute hors de la fin des lignes disponibles, comme après une opération FETCH ALL, le curseur est placé à gauche après la dernière ligne.

FORWARD 0 extrait la ligne actuelle sans déplacer le curseur ; autrement dit, extrait la ligne plus récemment extraite. Si le curseur est placé avant la première ligne ou après la dernière ligne, aucune ligne n'est retournée.

Lorsque la première ligne d'un curseur est extraite, le jeu complet de résultats est matérialisé sur le nœud principal, en mémoire ou sur disque, si nécessaire. En raison de l'impact négatif potentiel de l'utilisation de curseurs avec des ensembles de résultats volumineux sur les performances, nous recommandons autant que possible l'utilisation d'approches alternatives. Pour plus d'informations, consultez Considérations relatives aux performances lors de l'utilisation de curseurs.

Pour plus d'informations, consultez DECLARE, CLOSE.

Syntaxe

```
FETCH [ NEXT | ALL | {FORWARD [ count | ALL ] } ] FROM cursor
```

Paramètres

NEXT

Récupère la ligne suivante. Il s'agit de l'option par défaut.

ALL

Récupère toutes les lignes restantes. (Identique à FORWARD ALL.) ALL n'est pas pris en charge pour les clusters à un seul nœud.

FORWARD [nombre | ALL]

Extrait les nombre lignes suivantes, ou toutes les lignes restantes. `FORWARD 0` extrait la ligne actuelle. Pour les clusters à un seul nœud, la valeur maximale de count est `1000`. FORWARD ALL n'est pas pris en charge pour les clusters à un seul nœud.

curseur

Nom du nouveau schéma.

Exemple FETCH

L'exemple suivant déclare un curseur nommé LOLLAPALOOZA pour sélectionner les informations sur les ventes pour l'événement Lollapalooza, puis récupère les lignes de l'ensemble de résultats à l'aide du curseur :

```
-- Begin a transaction

begin;

-- Declare a cursor

declare lollapalooza cursor for
select eventname, starttime, pricepaid/qtysold as costperticket, qtysold
from sales, event
where sales.eventid = event.eventid
and eventname='Lollapalooza';

-- Fetch the first 5 rows in the cursor lollapalooza:

fetch forward 5 from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
--------------+---------------------+---------------+---------
 Lollapalooza | 2008-05-01 19:00:00 |   92.00000000 |       3
 Lollapalooza | 2008-11-15 15:00:00 |  222.00000000 |       2
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       3
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       4
 Lollapalooza | 2008-04-17 15:00:00 |  239.00000000 |       1
(5 rows)

-- Fetch the next row:
```

```
fetch next from lollapalooza;

  eventname   |      starttime      | costperticket | qtysold
--------------+---------------------+---------------+---------
 Lollapalooza | 2008-10-06 14:00:00 |  114.00000000 |       2

-- Close the cursor and end the transaction:

close lollapalooza;
commit;
```

GRANT

Définit les autorisations d'accès pour un utilisateur ou un rôle.

Les autorisations incluent les options d'accès telles que pouvoir lire les données des tables et des vues, écrire des données, créer et supprimer des tables. Utilisez cette commande pour accorder des autorisations spécifiques pour une table, une base de données, un schéma, une fonction, une procédure, un langage ou une colonne. Pour révoquer les autorisations d'un objet de base de données, utilisez la commande REVOKE.

Les autorisations incluent également les options d'accès producteur suivantes aux unités de partage des données :

- Octroi d'accès à l'unité de partage des données aux espaces de noms et comptes consommateur.
- Octroi de l'autorisation de modifier une unité de partage des données en y ajoutant ou supprimant des objets.
- Octroi de l'autorisation de partager une unité de partage des données en y ajoutant ou supprimant des espaces de noms consommateur.

Les options d'accès consommateur à l'unité de partage des données sont les suivantes :

- Octroi aux utilisateurs d'un accès complet aux bases de données créées à partir d'une unité de partage des données, ou à des schémas externes pointant vers de telles bases de données.
- Octroi aux utilisateurs des autorisations de niveau objet sur les bases de données créées à partir d'une unité de partage des données, comme cela est possible pour les objets de base de données locaux. Pour accorder ce niveau d'autorisation, vous devez utiliser la clause WITH PERMISSIONS lors de la création d'une base de données à partir de l'unité de partage des données. Pour de plus amples informations, veuillez consulter CREATE DATABASE.

Pour plus d'informations sur les autorisations liées aux unités de partage des données, consultez Autorisations que vous pouvez accorder aux unités de partages des données.

Vous pouvez également attribuer des rôles pour gérer les autorisations de la base de données et contrôler ce que les utilisateurs peuvent faire par rapport à vos données. En définissant des rôles et en les attribuant à des utilisateurs, vous pouvez limiter les actions que ces derniers peuvent entreprendre, par exemple en les limitant aux commandes CREATE TABLE et INSERT. Pour plus d'informations sur la commande CREATE ROLE, consultez the section called "CREATE ROLE". Amazon Redshift dispose de certains rôles définis par le système que vous pouvez également utiliser pour accorder des autorisations spécifiques à vos utilisateurs. Pour de plus amples informations, veuillez consulter the section called "Rôles définis par le système Amazon Redshift".

Vous pouvez accorder les autorisations GRANT ou REVOKE USAGE sur un schéma externe à des utilisateurs et des groupes d'utilisateurs de base de données qui utilisent la syntaxe ON SCHEMA. Lorsque vous utilisez ON EXTERNAL SCHEMA avec AWS Lake Formation, vous pouvez uniquement ACCORDER et RÉVOQUER des autorisations pour un rôle AWS Identity and Access Management (IAM). Pour connaître la liste des autorisations, reportez-vous à la syntaxe.

Pour les procédures stockées, la seule autorisation que vous pouvez accorder est EXECUTE.

Vous ne pouvez pas exécuter GRANT (sur une ressource externe) dans un bloc de transaction (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

Pour connaître les autorisations accordées aux utilisateurs pour une base de données, utilisez HAS_DATABASE_PRIVILEGE. Pour connaître les autorisations accordées aux utilisateurs pour un schéma, utilisez HAS_SCHEMA_PRIVILEGE. Pour connaître les autorisations accordées aux utilisateurs pour une table, utilisez HAS_TABLE_PRIVILEGE.

Syntaxe

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | DROP | REFERENCES | ALTER | TRUNCATE }
 [,...] | ALL [ PRIVILEGES ] }
    ON { [ TABLE ] table_name [, ...] | ALL TABLES IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { { CREATE | USAGE | TEMPORARY | TEMP | ALTER } [,...] | ALL [ PRIVILEGES ] }
    ON DATABASE db_name [, ...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

```
GRANT { { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }
    ON SCHEMA schema_name [, ...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON { FUNCTION function_name ( [ [ argname ] argtype [, ...] ] ) [, ...] | ALL
 FUNCTIONS IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON { PROCEDURE procedure_name ( [ [ argname ] argtype [, ...] ] ) [, ...] | ALL
 PROCEDURES IN SCHEMA schema_name [, ...] }
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT USAGE
    ON LANGUAGE language_name [, ...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { { ALTER | DROP} [,...] | ALL [ PRIVILEGES ] }
    ON COPY JOB job_name [,...]
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

Octroi d'autorisations au niveau des colonnes pour les tables

Voici la syntaxe des autorisations de niveau colonne sur les tables et les vues Amazon Redshift.

```
GRANT { { SELECT | UPDATE } ( column_name [, ...] ) [, ...] | ALL [ PRIVILEGES ]
 ( column_name [,...] ) }
     ON { [ TABLE ] table_name [, ...] }

     TO { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Octroi d'autorisations à ASSUMEROLE

Voici la syntaxe des autorisations ASSUMEROLE accordées aux utilisateurs et aux groupes ayant un rôle spécifié. Pour commencer à utiliser le privilège ASSUMEROLE, consultez Notes d'utilisation pour l'octroi de l'autorisation ASSUMEROLE.

```
GRANT ASSUMEROLE
       ON { 'iam_role' [, ...] | default | ALL }
       TO { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
       FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL } [, ...]
```

Octroi d'autorisations pour l'intégration de Redshift Spectrum avec Lake Formation

La syntaxe suivante s'applique à l'intégration de Redshift Spectrum à Lake Formation.

```
GRANT { SELECT | ALL [ PRIVILEGES ] } ( column_list )
    ON EXTERNAL TABLE schema_name.table_name
    TO { IAM_ROLE iam_role } [, ...] [ WITH GRANT OPTION ]

GRANT { { SELECT | ALTER | DROP | DELETE | INSERT }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL TABLE schema_name.table_name [, ...]
    TO { { IAM_ROLE iam_role } [, ...] | PUBLIC } [ WITH GRANT OPTION ]

GRANT { { CREATE | ALTER | DROP }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL SCHEMA schema_name [, ...]
    TO { IAM_ROLE iam_role } [, ...] [ WITH GRANT OPTION ]
```

Octroi d'autorisations d'unité de partage des données

Autorisations d'unité de partage des données côté producteur

La syntaxe suivante permet d'utiliser GRANT pour accorder des autorisations ALTER ou SHARE à un utilisateur ou à un rôle. L'utilisateur peut modifier l'unité de partage des données avec l'autorisation ALTER, ou en autoriser l'utilisation à un consommateur avec l'autorisation SHARE. ALTER et SHARE sont les seules autorisations que vous pouvez accorder aux utilisateurs et aux rôles sur une unité de partage des données.

```
GRANT { ALTER | SHARE } ON DATASHARE datashare_name
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

Voici la syntaxe d'utilisation de GRANT pour les autorisations d'unité de partage des données sur Amazon Redshift. Vous accordez l'accès à une unité de partage des données à un consommateur en utilisant l'autorisation USAGE. Vous ne pouvez pas accorder cette autorisation à des utilisateurs ou à des groupes d'utilisateurs. Cette autorisation ne prend pas non plus en charge WITH GRANT OPTION pour l'instruction GRANT. Seuls les utilisateurs ou groupes d'utilisateurs disposant de

l'autorisation SHARE qui leur a été préalablement accordée POUR l'unité de partage des données peuvent exécuter ce type d'instruction GRANT.

```
GRANT USAGE
    ON DATASHARE datashare_name
    TO NAMESPACE 'namespaceGUID' | ACCOUNT 'accountnumber' [ VIA DATA CATALOG ]
```

L'exemple suivant montre comment autoriser l'utilisation d'une unité de partage des données à un compte Lake Formation.

```
GRANT USAGE ON DATASHARE salesshare TO ACCOUNT '123456789012' VIA DATA CATALOG;
```

Autorisations d'unité de partage des données côté consommateur

Voici la syntaxe des autorisations d'utilisation d'unité de partage des données GRANT sur une base de données spécifique ou un schéma créé à partir d'une unité de partage des données.

Les autorisations supplémentaires requises pour que les consommateurs puissent accéder à une base de données créée à partir d'une unité de partage des données varient selon que la commande CREATE DATABASE utilisée pour créer la base de données à partir de l'unité de partage des données utilisait ou non la clause WITH PERMISSIONS. Pour plus d'informations sur la commande CREATE DATABASE et la clause WITH PERMISSIONS, consultez CREATE DATABASE.

Bases de données créées sans utiliser la clause WITH PERMISSIONS

Lorsque vous accordez l'accès USAGE sur une base de données créée à partir d'une unité de partage des données sans utiliser la clause WITH PERMISSIONS, vous n'avez pas besoin d'accorder des autorisations séparément sur les objets de la base de données partagée. Les entités autorisées à utiliser les bases de données créées à partir d'unités de partage des données sans la clause WITH PERMISSIONS ont automatiquement accès à tous les objets de la base de données.

Bases de données créées avec la clause WITH PERMISSIONS

Lorsque vous accordez l'accès USAGE à une base de données dont la base de données partagée a été créée à partir d'une unité de partage des données utilisant la clause WITH PERMISSIONS, les identités côté consommateur doivent toujours bénéficier des autorisations appropriées pour les objets de base de données de la base de données partagée afin de pouvoir y accéder, tout comme vous accorderiez des autorisations pour les objets de base de données locale. Pour accorder des autorisations aux objets d'une base de données créée à partir d'une unité de partage des données, utilisez la syntaxe en trois parties `database_name.schema_name.object_name`. Pour accorder

des autorisations aux objets d'un schéma externe pointant vers un schéma partagé au sein de la base de données partagée, utilisez la syntaxe en deux parties `schema_name.object_name`.

```
GRANT USAGE ON { DATABASE shared_database_name [, ...] | SCHEMA shared_schema}
    TO { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Octroi d'autorisations étendues

Les autorisations étendues vous permettent d'accorder des autorisations à un utilisateur ou à un rôle sur tous les objets d'un type au sein d'une base de données ou d'un schéma. Les utilisateurs et les rôles dotés d'autorisations étendues disposent des autorisations spécifiées sur tous les objets actuels et futurs de la base de données ou du schéma.

Vous pouvez consulter l'étendue des autorisations délimitées au niveau de la base de données dans. SVV_DATABASE_PRIVILEGES Vous pouvez consulter l'étendue des autorisations délimitées au niveau du schéma dans. SVV_SCHEMA_PRIVILEGES

Pour plus d'informations sur les autorisations délimitées, consultezAutorisations étendues.

La syntaxe suivante permet d'accorder des autorisations étendues aux utilisateurs et rôles.

```
GRANT { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }
FOR SCHEMAS IN
DATABASE db_name
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]

GRANT
{ { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [PRIVILEGES] } }
FOR TABLES IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO { username [ WITH GRANT OPTION ] | ROLE role_name} [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
FOR FUNCTIONS IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO { username [ WITH GRANT OPTION ] | ROLE role_name | } [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
FOR PROCEDURES IN
{SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
TO { username [ WITH GRANT OPTION ] | ROLE role_name | } [, ...]
```

```
GRANT USAGE
FOR LANGUAGES IN
{DATABASE db_name}
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]

GRANT { { CREATE | ALTER | DROP} [,...] | ALL [ PRIVILEGES ] }
FOR COPY JOBS
IN DATABASE db_name
TO { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]
```

Notez que les autorisations délimitées ne font pas de distinction entre les autorisations relatives aux fonctions et aux procédures. Par exemple, l'instruction suivante accorde bob l'EXECUTEautorisation pour les fonctions et les procédures du schémaSales_schema.

```
GRANT EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema TO bob;
```

Accorder des autorisations de machine learning

Voici la syntaxe pour les autorisations des modèles de machine learning sur Amazon Redshift.

```
GRANT CREATE MODEL
    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]

GRANT { EXECUTE | ALL [ PRIVILEGES ] }
    ON MODEL model_name [, ...]

    TO { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

Octroi d’autorisations de rôle

Voici la syntaxe permettant d'attribuer des rôles sur Amazon Redshift.

```
GRANT { ROLE role_name } [, ...] TO { { user_name [ WITH ADMIN OPTION ] } |
 ROLE role_name }[, ...]
```

Voici la syntaxe permettant d’accorder des autorisations système aux rôles sur Amazon Redshift. Notez que vous ne pouvez accorder des autorisations qu'aux rôles, et non aux utilisateurs.

```
GRANT
  {
```

```
    { CREATE USER | DROP USER | ALTER USER |
    CREATE SCHEMA | DROP SCHEMA |
    ALTER DEFAULT PRIVILEGES |
    ACCESS CATALOG | ACCESS SYSTEM TABLE
    CREATE TABLE | DROP TABLE | ALTER TABLE |
    CREATE OR REPLACE FUNCTION | CREATE OR REPLACE EXTERNAL FUNCTION |
    DROP FUNCTION |
    CREATE OR REPLACE PROCEDURE | DROP PROCEDURE |
    CREATE OR REPLACE VIEW | DROP VIEW |
    CREATE MODEL | DROP MODEL |
    CREATE DATASHARE | ALTER DATASHARE | DROP DATASHARE |
    CREATE LIBRARY | DROP LIBRARY |
    CREATE ROLE | DROP ROLE |
    TRUNCATE TABLE
    VACUUM | ANALYZE | CANCEL |
    IGNORE RLS | EXPLAIN RLS |
    EXPLAIN MASKING }[, ...]
  }
  | { ALL [ PRIVILEGES ] }
TO ROLE role_name [, ...]
```

Octroi d'autorisations d'explication pour les filtres de politique de sécurité au niveau des lignes

Voici la syntaxe permettant d'accorder des autorisations pour expliquer les filtres de politique de sécurité au niveau des lignes d'une requête dans le plan EXPLAIN. Vous pouvez révoquer ce privilège à l'aide de l'instruction REVOKE.

```
GRANT EXPLAIN RLS TO ROLE rolename
```

Voici la syntaxe permettant d'accorder des autorisations permettant de contourner les politiques de sécurité au niveau des lignes pour une requête.

```
GRANT IGNORE RLS TO ROLE rolename
```

Octroi d'autorisations pour les tables de recherche RLS à un objet de politique

Voici la syntaxe permettant d'accorder des autorisations à la politique de sécurité au niveau des lignes spécifiée.

```
GRANT SELECT ON [ TABLE ] table_name [, ...]
TO RLS POLICY policy_name [, ...]
```

Paramètres

SELECT

Autorise la sélection de données dans une table ou une vue à l'aide d'une instruction SELECT. L'autorisation SELECT est également requise pour référencer les valeurs des colonnes existantes pour les opérations UPDATE ou DELETE.

INSERT

Autorise le chargement de données dans une table à l'aide d'une instruction INSERT ou d'une instruction COPY.

UPDATE

Autorise la mise à jour d'une colonne de table à l'aide d'une instruction UPDATE. Les opérations UPDATE nécessitent également l'autorisation SELECT, car elles doivent faire référence aux colonnes de la table pour déterminer les lignes à mettre à jour ou pour calculer les nouvelles valeurs des colonnes.

DELETE

Autorise la suppression d'une ligne de données d'une table. Les opérations DELETE nécessitent également l'autorisation SELECT, car elles doivent faire référence aux colonnes de la table pour déterminer les lignes à supprimer.

DROP

En fonction de l'objet de base de données, accorde les autorisations suivantes à l'utilisateur ou au rôle :

- Pour les tables, DROP autorise la suppression d'une table ou d'une vue. Pour de plus amples informations, veuillez consulter DROP TABLE.
- Pour les bases de données, DROP autorise la suppression d'une base de données. Pour de plus amples informations, veuillez consulter DROP DATABASE.
- Pour les schémas, DROP autorise la suppression d'un schéma. Pour de plus amples informations, veuillez consulter DROP SCHEMA.

REFERENCES

Autorise la création d'une contrainte de clé étrangère. Vous devez accorder cette autorisation à la fois à la table référencée et à la table qui fait référence ; sinon, l'utilisateur ne peut pas créer la contrainte.

ALTER

En fonction de l'objet de la base de données, accorde les autorisations suivantes à l'utilisateur ou au groupe d'utilisateurs :

- Pour les tables, ALTER autorise la modification d'une table ou d'une vue. Pour de plus amples informations, veuillez consulter ALTER TABLE.
- Pour les bases de données, ALTER autorise la modification d'une base de données. Pour de plus amples informations, veuillez consulter ALTER DATABASE.
- Pour les schémas, ALTER autorise la modification d'un schéma. Pour de plus amples informations, veuillez consulter ALTER SCHEMA.
- Pour les tables externes, ALTER autorise la modification d'une table dans un AWS Glue Data Catalog qui est activé pour Lake Formation. Cette autorisation s'applique uniquement lors de l'utilisation de Lake Formation.

TRUNCATE

Accorde l'autorisation de tronquer une table. Sans cette autorisation, seul le propriétaire d'une table ou un super-utilisateur peut tronquer une table. Pour plus d'informations sur la commande TRUNCATE, consultez the section called "TRUNCATE".

ALL [PRIVILEGES]

Accorde toutes les autorisations disponibles en une seule fois à l'utilisateur ou au rôle spécifié. Le mot-clé PRIVILEGES est facultatif.

GRANT ALL ON SCHEMA n'accorde pas les autorisations CREATE pour les schémas externes.

Vous pouvez accorder l'autorisation ALL à une table dans un AWS Glue Data Catalog fichier activé pour Lake Formation. Dans ce cas, les autorisations individuelles (telles que SELECT, ALTER, etc.) sont enregistrées dans le catalogue de données.

> **Note**
>
> Amazon Redshift ne prend pas en charge les autorisations RULE et TRIGGER. Pour plus d'informations, consultez la section concernant Fonctions PostgreSQL non prises en charge.

ASSUMEROLE

Accorde l'autorisation d'exécuter les commandes COPY, UNLOAD, EXTERNAL FUNCTION et CREATE MODEL aux utilisateurs, aux rôles et aux groupes ayant un rôle spécifié. L'utilisateur, le rôle ou le groupe assume ce rôle lors de l'exécution de la commande spécifiée. Pour commencer à utiliser l'autorisation ASSUMEROLE, consultez Notes d'utilisation pour l'octroi de l'autorisation ASSUMEROLE.

ON [TABLE] nom_table

Accorde les autorisations spécifiées sur une table ou une vue. Le mot-clé TABLE est facultatif. Vous pouvez afficher plusieurs tables et vues dans une seule instruction.

ON ALL TABLES IN SCHEMA nom_schéma

Accorde les autorisations spécifiées sur toutes les tables et vues du schéma référencé.

(nom_colonne [,...]) ON TABLE nom_table

Accorde les autorisations spécifiées aux utilisateurs, groupes ou PUBLIC sur les colonnes spécifiées de la table ou vue Amazon Redshift.

(column_list) ON EXTERNAL TABLE schema_name.table_name

Accorde les autorisations spécifiées à un rôle IAM sur les colonnes spécifiées de la table Lake Formation dans le schéma référencé.

ON EXTERNAL TABLE schema_name.table_name

Accorde les autorisations spécifiées à un rôle IAM sur les tables Lake Formation spécifiées dans le schéma référencé.

ON EXTERNAL SCHEMA schema_name

Accorde les autorisations spécifiées à un rôle IAM sur le schéma référencé.

ON iam_role

Accorde les autorisations spécifiées à un rôle IAM.

TO nom_utilisateur

Indique l'utilisateur qui reçoit les autorisations.

TO IAM_ROLE iam_role

Indique le rôle IAM qui reçoit les autorisations.

WITH GRANT OPTION

Indique que l'utilisateur qui reçoit les autorisations peut à son tour accorder les mêmes autorisations à d'autres personnes. WITH GRANT OPTION ne peut pas être accordé à un groupe ou à PUBLIC.

ROLE role_name

Octroie les autorisations à un rôle.

GROUP group_name

Octroie les autorisations à un groupe d'utilisateurs. Il peut s'agir d'une liste séparée par des virgules pour spécifier plusieurs groupes d'utilisateurs.

PUBLIC

Accorde les autorisations spécifiées à tous les utilisateurs, y compris aux utilisateurs créés ultérieurement. PUBLIC représente un groupe qui inclut toujours tous les utilisateurs. Les autorisations d'un utilisateur sont la somme des autorisations accordées à PUBLIC, des autorisations accordées aux groupes auxquels l'utilisateur appartient et des autorisations accordées à l'utilisateur à titre individuel.

L'octroi de PUBLIC à une TABLE EXTERNE de Lake Formation a pour effet d'accorder l'autorisation au groupe de personnes tout le monde de Lake Formation.

CREATE

En fonction de l'objet de la base de données, accorde les autorisations suivantes à l'utilisateur ou au groupe d'utilisateurs :

- Pour les bases de données, CREATE permet aux utilisateurs de créer des schémas au sein de la base de données.
- Pour les schémas, CREATE permet aux utilisateurs de créer des objets au sein d'un schéma. Pour renommer un objet, l'utilisateur doit disposer de l'autorisation CREATE et posséder l'objet à renommer.
- CREATE ON SCHEMA n'est pas pris en charge pour les schémas externes Amazon Redshift Spectrum. Pour accorder l'utilisation de tables externes dans un schéma externe, accordez USAGE ON SCHEMA aux utilisateurs qui nécessitent un accès. Seul le propriétaire d'un schéma externe ou un super-utilisateur est autorisé à créer des tables externes dans le schéma externe. Pour transférer la propriété d'un schéma externe, utilisez ALTER SCHEMA pour modifier le propriétaire.

TEMPORARY | TEMP

Accorde l'autorisation de créer des tables temporaires dans la base de données spécifiée. Pour exécuter des requêtes Amazon Redshift Spectrum, l'utilisateur de la base de données doit avoir l'autorisation d'y créer des tables temporaires.

> **Note**
>
> Par défaut, les utilisateurs ont l'autorisation de créer des tables temporaires grâce à leur appartenance automatique au groupe PUBLIC. Pour retirer à tout utilisateur l'autorisation de créer des tables temporaires, révoquez l'autorisation TEMP du groupe PUBLIC. Accordez ensuite explicitement l'autorisation de créer des tables temporaires à des utilisateurs ou à des groupes d'utilisateurs spécifiques.

ON DATABASE nom_db

Accorde les autorisations spécifiées à une base de données.

USAGE

Accorde l'autorisation USAGE sur un schéma spécifique, ce qui rend les objets de ce schéma accessibles aux utilisateurs. Les actions spécifiques sur ces objets doivent être accordées séparément (par exemple, l'autorisation SELECT ou UPDATE sur les tables) pour les schémas Amazon Redshift locaux. Par défaut, tous les utilisateurs disposent de l'autorisation CREATE et USAGE sur le schéma PUBLIC.

Lorsque vous autorisez USAGE pour des schémas externes à l'aide de la syntaxe ON SCHEMA, vous n'avez pas besoin d'accorder des actions séparément sur les objets du schéma externe. Les autorisations de catalogue correspondantes contrôlent les autorisations granulaires sur les objets de schéma externes.

ON SCHEMA nom_schéma

Accorde les autorisations spécifiées à un schéma.

GRANT CREATE ON SCHEMA et l'autorisation CREATE dans GRANT ALL ON SCHEMA ne sont pas pris en charge pour les schémas externes Amazon Redshift Spectrum. Pour accorder l'utilisation de tables externes dans un schéma externe, accordez USAGE ON SCHEMA aux utilisateurs qui nécessitent un accès. Seul le propriétaire d'un schéma externe ou un super-

utilisateur est autorisé à créer des tables externes dans le schéma externe. Pour transférer la propriété d'un schéma externe, utilisez ALTER SCHEMA pour modifier le propriétaire.

EXECUTE ON ALL FUNCTIONS IN SCHEMA nom_schéma

Accorde les autorisations spécifiées à toutes les fonctions du schéma référencé.

Amazon Redshift ne prend pas en charge les instructions GRANT ou REVOKE pour les entrées intégrées pg_proc définies dans l'espace de noms pg_catalog.

EXECUTE ON PROCEDURE procedure_name

Accorde l'autorisation EXECUTE à une procédure stockée spécifique. Comme les noms de procédures stockées peuvent être surchargés, vous devez inclure la liste des arguments de la procédure. Pour plus d'informations, consultez Dénomination des procédures stockées.

EXECUTE ON ALL PROCEDURES IN SCHEMA nom_schéma

Accorde les autorisations spécifiées à toutes les procédures stockées du schéma référencé.

USAGE ON LANGUAGE nom_langage

Accorde l'autorisation USAGE à une langue.

L'autorisation USAGE ON LANGUAGE est requise pour créer des fonctions définies par l'utilisateur (UDFs) en exécutant la CREATE FUNCTION commande. Pour de plus amples informations, veuillez consulter Sécurité et autorisations UDF.

L'autorisation USAGE ON LANGUAGE est requise pour créer des procédures stockées définies par l'utilisateur (UDF) en exécutant la commande CREATE PROCEDURE. Pour de plus amples informations, veuillez consulter Sécurité et privilèges des procédures stockées .

Pour Python UDFs, utilisez`plpythonu`. Pour SQL UDFs, utilisez`sql`. Pour les procédures stockées, utilisez `plpgsql`.

ON COPY JOB job_name

Accorde les autorisations spécifiées pour une tâche de copie.

FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL } [, ...]

Spécifie la commande SQL pour laquelle l'autorisation est accordée. Vous pouvez spécifier ALL pour accorder l'autorisation sur les instructions COPY, UNLOAD, EXTERNAL FUNCTION et CREATE MODEL. Cette clause ne s'applique qu'à l'octroi de l'autorisation ASSUMEROLE.

ALTER

Accorde l'autorisation ALTER aux utilisateurs pour ajouter ou supprimer des objets d'une unité de partage des données, ou pour définir la propriété PUBLICACCESSIBLE. Pour de plus amples informations, veuillez consulter ALTER DATASHARE.

SHARE

Accorde des autorisations aux utilisateurs et aux groupes d'utilisateurs pour ajouter des consommateurs de données à une unité de partage des données. Cette autorisation est requise pour permettre au consommateur particulier (compte ou espace de noms) d'accéder à l'unité de partage des données à partir de ses clusters. Le consommateur peut être le même compte ou un AWS compte différent, avec le même espace de noms de cluster ou un autre, tel que spécifié par un identifiant unique global (GUID).

ON DATASHARE datashare_name

Accorde les autorisations spécifiées sur l'unité de partage des données référencée. Pour obtenir des informations sur la granularité du contrôle d'accès des consommateurs, consultez Partage de données à différents niveaux dans Amazon Redshift.

USAGE

Lorsque le privilège USAGE est accordé à un compte consommateur ou à un espace de noms au sein du même compte, le compte ou l'espace de noms spécifique du compte peut accéder à l'unité de partage des données et aux objets de l'unité de partage des données en lecture seule.

TO NAMESPACE 'clusternamespace GUID'

Indique un espace de noms dans le même compte où les consommateurs peuvent recevoir les autorisations spécifiées pour l'unité de partage des données. Les espaces de noms utilisent un GUID alphanumérique 128 bits.

TO ACCOUNT 'accountnumber' [VIA DATA CATALOG]

Indique le numéro d'un autre compte dont les consommateurs peuvent recevoir les autorisations spécifiées pour l'unité de partage des données. La spécification « VIA DATA CATALOG » indique que vous autorisez un compte Lake Formation à utiliser l'unité de partage des données. L'omission de ce paramètre signifie que vous accordez l'utilisation à un compte propriétaire du cluster.

ON DATABASE shared_database_name> [, ...]

Accorde les autorisations d'utilisation spécifiées à la base de données spécifiée qui est créée dans l'unité de partage des données spécifiée.

ON SCHEMA shared_schema

Accorde les autorisations spécifiées au schéma spécifié qui est créé dans l'unité de partage des données spécifiée.

POUR {SCHÉMAS | TABLES | FONCTIONS | PROCÉDURES | LANGUES | TÂCHES DE COPIE} DANS

Spécifie les objets de base de données auxquels accorder une autorisation. Les paramètres situés après IN définissent l'étendue de l'autorisation accordée.

CREATE MODEL

Accorde l'autorisation CREATE MODEL à des utilisateurs ou groupes d'utilisateurs spécifiques.

ON MODEL model_name

Accorde l'autorisation EXECUTE à un modèle spécifique.

ACCESS CATALOG

Accorde l'autorisation d'afficher les métadonnées pertinentes des objets auxquels le rôle a accès.

{ role } [, ...]

Rôle à attribuer à un autre rôle, à un utilisateur ou à PUBLIC.

PUBLIC représente un groupe qui inclut toujours tous les utilisateurs. Les autorisations d'un utilisateur sont la somme des autorisations accordées à PUBLIC, des autorisations accordées aux groupes auxquels l'utilisateur appartient et des autorisations accordées à l'utilisateur à titre individuel.

TO { { user_name [WITH ADMIN OPTION] } | role }[, ...]

Attribue le rôle spécifié à un utilisateur spécifié avec WITH ADMIN OPTION, à un autre rôle ou à PUBLIC.

La clause WITH ADMIN OPTION fournit les options d'administration de tous les rôles accordés à tous les bénéficiaires.

EXPLAIN RLS TO ROLE rolename

Accorde à un rôle l'autorisation d'expliquer les filtres de politique de sécurité au niveau des lignes d'une requête dans le plan EXPLAIN.

IGNORE RLS TO ROLE rolename

Accorde à un rôle l'autorisation de contourner les politiques de sécurité au niveau des lignes pour une requête.

Notes d'utilisation

Pour en savoir plus sur les notes d'utilisation de GRANT, consultez the section called "Notes d'utilisation".

Exemples

Pour des exemples d'utilisation de GRANT, consultez the section called "Exemples".

Notes d'utilisation

Pour attribuer des privilèges sur un objet, vous devez répondre à l'un des critères suivants :

- Être le propriétaire de l'objet.
- Être un super-utilisateur.
- Avoir un privilège accordé pour cet objet et ce privilège.

Par exemple, la commande suivante autorise l'utilisateur HR à exécuter des commandes SELECT sur la table des employés et à accorder et révoquer le même privilège aux autres utilisateurs.

```
grant select on table employees to HR with grant option;
```

HR ne peut pas accorder de privilèges pour une opération autre que SELECT, ou sur toute autre table que celle des employés.

Par exemple, la commande suivante autorise l'utilisateur HR à exécuter des commandes ALTER sur la table des employés et à accorder et révoquer le même privilège aux autres utilisateurs.

```
grant ALTER on table employees to HR with grant option;
```

HR ne peut pas accorder de privilèges pour une opération autre que ALTER ou sur toute autre table que celle des employés.

Le fait d'avoir les privilèges octroyés sur une vue n'implique pas d'avoir les privilèges sur les tables sous-jacentes. De même, le fait d'avoir les privilèges octroyés sur un schéma n'implique pas d'avoir les privilèges sur les tables du schéma. Accordez explicitement l'accès aux tables sous-jacentes.

Pour accorder des privilèges à une AWS Lake Formation table, le rôle IAM associé au schéma externe de la table doit être autorisé à accorder des privilèges à la table externe. L'exemple suivant crée un schéma externe avec un rôle IAM `myGrantor` associé. Le rôle IAM `myGrantor` a l'autorisation d'accorder des autorisations. La commande GRANT utilise l'autorisation du rôle IAM `myGrantor` associé au schéma externe pour accorder des autorisations au rôle IAM `myGrantee`.

```
create external schema mySchema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myGrantor'
create external database if not exists;
```

```
grant select
on external table mySchema.mytable
to iam_role 'arn:aws:iam::123456789012:role/myGrantee';
```

Si vous accordez des privilèges GRANT ALL à un rôle IAM, des privilèges individuels sont accordés dans le catalogue de données Lake Formation. Par exemple, les privilèges GRANT ALL suivants permettent aux privilèges individuels accordés (SELECT, ALTER, DROP, DELETE et INSERT) de s'afficher dans la console Lake Formation.

```
grant all
on external table mySchema.mytable
to iam_role 'arn:aws:iam::123456789012:role/myGrantee';
```

Les super-utilisateurs peuvent accéder à tous les objets, quelle que soit les commandes GRANT et REVOKE qui définissent les privilèges d'objet.

Notes d'utilisation pour le contrôle d'accès de niveau colonne

Les notes d'utilisation suivantes s'appliquent aux privilèges de niveau colonne sur les tables et les vues Amazon Redshift. Ces notes décrivent des tableaux ; les mêmes notes s'appliquent aux vues, sauf signalement explicite d'une exception.

- Pour une table Amazon Redshift, vous pouvez accorder uniquement les privilèges SELECT et UPDATE au niveau de la colonne. Pour une vue Amazon Redshift, vous pouvez accorder uniquement le privilège SELECT au niveau de la colonne.
- Le mot-clé ALL est synonyme des privilèges combinés SELECT et UPDATE lorsqu'ils sont utilisés dans le contexte d'un GRANT de niveau colonne sur une table.
- Si vous ne disposez pas du privilège SELECT sur toutes les colonnes d'une table, l'opération SELECT * ne renvoie que les colonnes auxquelles vous avez accès. Lorsque vous utilisez une vue, une opération SELECT * tente d'accéder à toutes les colonnes de la vue. Si vous n'êtes pas autorisé à accéder à toutes les colonnes, ces requêtes échouent avec un message d'erreur de refus d'autorisation.
- SELECT * ne s'étend pas aux seules colonnes accessibles dans les cas suivants :
  - Vous ne pouvez pas créer une vue normale avec seulement des colonnes accessibles en utilisant SELECT *.
  - Vous ne pouvez pas créer une vue matérialisée avec seulement des colonnes accessibles en utilisant SELECT *.
- Si vous disposez du privilège SELECT ou UPDATE sur une table ou une vue et que vous ajoutez une colonne, vous disposez toujours des mêmes privilèges sur la table ou la vue et donc, sur toutes ses colonnes.
- Seul le propriétaire d'une table ou un superutilisateur peut accorder des privilèges de niveau colonne.
- La clause WITH GRANT OPTION n'est pas prise en charge pour les privilèges de niveau colonne.
- Vous ne pouvez pas détenir le même privilège au niveau de la table et de la colonne. Par exemple, l'utilisateur `data_scientist` ne peut pas avoir à la fois le privilège SELECT sur la table `employee` et le privilège SELECT sur la colonne `employee.department`. Tenez compte des résultats suivants lorsque vous accordez le même privilège à une table et à une colonne de la table :
  - Si un utilisateur dispose d'un privilège de niveau table sur une table, l'octroi du même privilège au niveau de la colonne n'a aucun effet.
  - Si un utilisateur dispose d'un privilège de niveau table sur une table, la révocation du même privilège pour une ou plusieurs colonnes de la table renvoie une erreur. Au lieu de cela, révoquez le privilège au niveau de la table.
  - Si un utilisateur dispose d'un privilège au niveau de la colonne, l'octroi du même privilège au niveau de la table renvoie une erreur.

  - Si un utilisateur dispose d'un privilège au niveau de la colonne, la révocation du même privilège au niveau de la table révoque les privilèges de colonne et de table pour toutes les colonnes de la table.
- Vous ne pouvez pas accorder de privilèges de niveau colonne sur les vues de liaison tardive.
- Pour créer une vue matérialisée, vous devez disposer du privilège SELECT au niveau de la table sur les tables de base. Même si vous disposez de privilèges au niveau de colonnes spécifiques, vous ne pouvez pas créer de vue matérialisée uniquement avec ces colonnes. Toutefois, vous pouvez accorder le privilège SELECT aux colonnes d'une vue matérialisée, de la même manière que les vues régulières.
- Pour rechercher des privilèges de niveau colonne, utilisez la vue PG_ATTRIBUTE_INFO.

Notes d'utilisation pour l'octroi de l'autorisation ASSUMEROLE

Les notes d'utilisation suivantes s'appliquent à l'octroi de l'autorisation ASSUMEROLE dans Amazon Redshift.

Vous utilisez l'autorisation ASSUMEROLE pour contrôler les autorisations d'accès des rôles IAM pour les utilisateurs, les rôles ou les groupes de bases de données sur des commandes telles que COPY, UNLOAD, EXTERNAL FUNCTION ou CREATE MODEL. Après avoir accordé l'autorisation ASSUMEROLE à un utilisateur, un rôle ou un groupe pour un rôle IAM, l'utilisateur, le rôle ou le groupe peut assumer ce rôle lors de l'exécution de la commande. L'autorisation ASSUMEROLE vous permet d'accorder l'accès aux autorisations appropriées en fonction des besoins.

Seul un super-utilisateur de la base de données peut accorder ou révoquer l'autorisation ASSUMEROLE pour les utilisateurs, les rôles et les groupes. Un super-utilisateur conserve toujours l'autorisation ASSUMEROLE.

Pour autoriser l'utilisation de l'autorisation ASSUMEROLE pour les utilisateurs, les rôles et les groupes, un super-utilisateur effectue les deux actions suivantes :

- Exécuter l'instruction suivante une fois sur le cluster :

```
revoke assumerole on all from public for all;
```

- Accordez l'autorisation ASSUMEROLE aux utilisateurs, aux rôles et aux groupes pour les autorisations appropriées.

Vous pouvez spécifier la création de chaînes de rôles dans la clause ON lorsque vous accordez l'autorisation ASSUMEROLE. Vous utilisez des virgules pour séparer les rôles dans une chaîne de rôles, par exemple `Role1,Role2,Role3`. Si la création de chaînes de rôles a été spécifiée lors de l'octroi de l'autorisation ASSUMEROLE, vous devez spécifier la chaîne de rôles lors de l'exécution des autorisations accordées par l'autorisation ASSUMEROLE. Vous ne pouvez pas spécifier des rôles individuels dans la chaîne de rôles lorsque vous exécutez des autorisations accordées par l'autorisation ASSUMEROLE. Par exemple, si un utilisateur, un rôle ou un groupe se voit attribuer la chaîne de rôles `Role1,Role2,Role3`, vous ne pouvez pas spécifier uniquement `Role1` pour effectuer des opérations.

Si un utilisateur tente d'effectuer une opération COPY, UNLOAD, EXTERNAL FUNCTION ou CREATE MODEL et n'a pas obtenu l'autorisation ASSUMEROLE, un message semblable au suivant s'affiche.

```
ERROR:  User awsuser does not have ASSUMEROLE permission on IAM role
 "arn:aws:iam::123456789012:role/RoleA" for COPY
```

Pour répertorier les utilisateurs qui ont obtenu l'accès aux rôles et commandes IAM via l'autorisation ASSUMEROLE, consultez HAS_ASSUMEROLE_PRIVILEGE. Pour répertorier les rôles IAM et les autorisations de commande qui ont été accordées à un utilisateur que vous spécifiez, consultez PG_GET_IAM_ROLE_BY_USER. Pour répertorier les utilisateurs, les rôles et les groupes auxquels l'accès a été accordé à un rôle IAM que vous spécifiez, consultez PG_GET_GRANTEE_BY_IAM_ROLE.

Notes d'utilisation pour l'octroi d'autorisations de machine learning

Vous ne pouvez pas accorder ou révoquer directement les autorisations liées à une fonction ML. Une fonction ML appartient à un modèle ML et les autorisations sont contrôlées par le biais du modèle. En revanche, vous pouvez accorder des autorisations liées au modèle ML. L'exemple suivant montre comment autoriser tous les utilisateurs à exécuter la fonction ML associée au modèle `customer_churn`.

```
GRANT EXECUTE ON MODEL customer_churn TO PUBLIC;
```

Vous pouvez également accorder toutes les autorisations à un utilisateur pour le modèle ML`customer_churn`.

```
GRANT ALL on MODEL customer_churn TO ml_user;
```

L'octroi de l'autorisation EXECUTE liée à une fonction de ML échouera s'il existe une fonction de ML dans le schéma, même si cette fonction de ML dispose déjà de l'autorisation EXECUTE par le biais de GRANT EXECUTE ON MODEL. Nous vous recommandons d'utiliser un schéma distinct lorsque vous utilisez la commande CREATE MODEL afin de conserver les fonctions ML dans un schéma distinct. L'exemple suivant vous montre comment procéder.

```
CREATE MODEL ml_schema.customer_churn
FROM customer_data
TARGET churn
FUNCTION ml_schema.customer_churn_prediction
IAM_ROLE default
SETTINGS (
 S3_BUCKET 'amzn-s3-demo-bucket'
);
```

Exemples

L'exemple suivant accorde le privilège SELECT sur la table SALES à l'utilisateur fred.

```
grant select on table sales to fred;
```

L'exemple suivant accorde le privilège SELECT sur toutes les tables du schéma QA_TICKIT à l'utilisateur fred.

```
grant select on all tables in schema qa_tickit to fred;
```

L'exemple suivant accorde tous les privilèges de schéma sur le schéma QA_TICKIT au groupe d'utilisateurs QA_USERS. Les privilèges de schéma sont CREATE et USAGE. USAGE accorde aux utilisateurs l'accès aux objets du schéma, mais n'accorde pas les privilèges tels que INSERT or SELECT sur ces objets. Accordez des privilèges sur chaque objet séparément.

```
create group qa_users;
grant all on schema qa_tickit to group qa_users;
```

L'exemple suivant accorde tous les privilèges sur la table SALES du schéma QA_TICKIT à tous les utilisateurs du groupe QA_USERS.

```
grant all on table qa_tickit.sales to group qa_users;
```

L'exemple suivant accorde tous les privilèges sur la table SALES du schéma QA_TICKIT à tous les utilisateurs des groupes QA_USERS et RO_USERS.

```
grant all on table qa_tickit.sales to group qa_users, group ro_users;
```

L'exemple suivant accorde le privilège DROP sur la table SALES du schéma QA_TICKIT à tous les utilisateurs du groupe QA_USERS.

```
grant drop on table qa_tickit.sales to group qa_users;>
```

La séquence de commandes suivante montre comment l'accès à un schéma n'accorde pas de privilèges sur une table du schéma.

```
create user schema_user in group qa_users password 'Abcd1234';
create schema qa_tickit;
create table qa_tickit.test (col1 int);
grant all on schema qa_tickit to schema_user;

set session authorization schema_user;
select current_user;


current_user
--------------
schema_user
(1 row)


select count(*) from qa_tickit.test;


ERROR: permission denied for relation test [SQL State=42501]


set session authorization dw_user;
grant select on table qa_tickit.test to schema_user;
set session authorization schema_user;
select count(*) from qa_tickit.test;


count
-------
```

```
0
(1 row)
```

La séquence de commandes suivante montre comment l'accès à une vue n'implique pas l'accès à ses tables sous-jacentes. L'utilisateur appelé VIEW_USER ne peut pas sélectionner à partir de la table DATE, même si cet utilisateur s'est vu accorder tous les privilèges sur VIEW_DATE.

```
create user view_user password 'Abcd1234';
create view view_date as select * from date;
grant all on view_date to view_user;
set session authorization view_user;
select current_user;


current_user
--------------
view_user
(1 row)


select count(*) from view_date;


count
-------
365
(1 row)


select count(*) from date;


ERROR:  permission denied for relation date
```

L'exemple suivant accorde le privilège SELECT sur les colonnes `cust_name` et `cust_phone` de la table `cust_profile` à l'utilisateur `user1`.

```
grant select(cust_name, cust_phone) on cust_profile to user1;
```

L'exemple suivant accorde le privilège SELECT sur les colonnes cust_name et cust_phone et le privilège UPDATE sur la colonne cust_contact_preference de la table cust_profile au groupe sales_group.

```
grant select(cust_name, cust_phone), update(cust_contact_preference) on cust_profile to
 group sales_group;
```

L'exemple suivant montre l'utilisation du mot-clé ALL pour accorder au groupe sales_admin les privilèges SELECT et UPDATE sur trois colonnes de la table cust_profile.

```
grant ALL(cust_name, cust_phone,cust_contact_preference) on cust_profile to group
 sales_admin;
```

L'exemple suivant accorde à l'utilisateur user2 le privilège SELECT sur la colonne cust_name de la vue cust_profile_vw.

```
grant select(cust_name) on cust_profile_vw to user2;
```

Exemples d'octroi d'accès à des unités de partage des données

Les exemples suivants illustrent les autorisations d'utilisation de partage de données GRANT sur une base de données spécifique ou un schéma créé à partir d'une unité de partage des données.

Dans l'exemple suivant, un administrateur côté producteur accorde à l'espace de noms spécifié l'autorisation USAGE sur l'unité de partage des données salesshare.

```
GRANT USAGE ON DATASHARE salesshare TO NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

Dans l'exemple suivant, un administrateur côté consommateur accorde à Bob l'autorisation USAGE sur la base de données sales_db.

```
GRANT USAGE ON DATABASE sales_db TO Bob;
```

Dans l'exemple suivant, un administrateur côté consommateur accorde au rôle Analyst_rolel'autorisation GRANT USAGE sur le schéma sales_schema. sales_schema est un schéma externe qui pointe vers sales_db.

```
GRANT USAGE ON SCHEMA sales_schema TO ROLE Analyst_role;
```

À ce stade, Bob et Analyst_role peuvent accéder à tous les objets de base de données dans sales_schema etsales_db.

L'exemple suivant montre comment accorder une autorisation de niveau objet supplémentaire pour les objets d'une base de données partagée. Ces autorisations supplémentaires ne sont nécessaires que si la commande CREATE DATABASE utilisée pour créer la base de données partagée utilisait la clause WITH PERMISSIONS. Si la commande CREATE DATABASE n'a pas utilisé la clause WITH PERMISSIONS, accorder l'autorisation USAGE sur la base de données partagée donne un accès complet à tous les objets de cette base de données.

```
GRANT SELECT ON sales_db.sales_schema.tickit_sales_redshift to Bob;
```

Exemples d'octroi d'autorisations étendues

L'exemple suivant autorise le rôle Sales à utiliser tous les schémas actuels et futurs de la base de données Sales_db.

```
GRANT USAGE FOR SCHEMAS IN DATABASE Sales_db TO ROLE Sales;
```

L'exemple suivant accorde à l'utilisateur alice l'autorisation SELECT pour toutes les tables actuelles et futures de la base de données Sales_db, et donne aussi à alice l'autorisation d'accorder à d'autres utilisateurs des autorisations étendues sur les tables de Sales_db.

```
GRANT SELECT FOR TABLES IN DATABASE Sales_db TO alice WITH GRANT OPTION;
```

L'exemple suivant accorde à l'utilisateur bob l'autorisation EXECUTE pour les fonctions du schéma Sales_schema.

```
GRANT EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema TO bob;
```

L'exemple suivant accorde au Sales rôle toutes les autorisations pour toutes les tables du schéma ShareSchema de la base de données ShareDb. Lorsque vous spécifiez le schéma, vous pouvez spécifier la base de données du schéma en utilisant le format en deux parties database.schema.

```
GRANT ALL FOR TABLES IN SCHEMA ShareDb.ShareSchema TO ROLE Sales;
```

L'exemple suivant est le même que le précédent. Vous pouvez spécifier la base de données à l'aide du mot-clé DATABASE au lieu d'utiliser un format en deux parties.

```
GRANT ALL FOR TABLES IN SCHEMA ShareSchema DATABASE ShareDb TO ROLE Sales;
```

Exemples d'octroi du privilège ASSUMEROLE

Voici des exemples d'octroi du privilège ASSUMEROLE.

L'exemple suivant montre l'instruction REVOKE qu'un super-utilisateur exécute une fois sur le cluster pour activer l'utilisation du privilège ASSUMEROLE pour les utilisateurs et les groupes. Ensuite, le super-utilisateur accorde le privilège ASSUMEROLE aux utilisateurs et aux groupes pour les commandes appropriées. Pour obtenir des informations sur l'activation du privilège ASSUMEROLE pour les utilisateurs et les groupes, consultez Notes d'utilisation pour l'octroi de l'autorisation ASSUMEROLE.

```
revoke assumerole on all from public for all;
```

L'exemple suivant accorde le privilège ASSUMEROLE à l'utilisateur `reg_user1` pour le rôle IAM `Redshift-S3-Read` pour effectuer des opérations COPY.

```
grant assumerole on 'arn:aws:iam::123456789012:role/Redshift-S3-Read'
to reg_user1 for copy;
```

L'exemple suivant accorde le privilège ASSUMEROLE à l'utilisateur `reg_user1` pour la chaîne de rôles IAM `RoleA`, `RoleB` pour effectuer des opérations UNLOAD.

```
grant assumerole
on 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB'
to reg_user1
for unload;
```

Voici un exemple de commande UNLOAD utilisant la chaîne de rôles IAM `RoleA`, `RoleB`.

```
unload ('select * from venue limit 10')
to 's3://companyb/redshift/venue_pipe_'
iam_role 'arn:aws:iam::123456789012:role/RoleA,arn:aws:iam::210987654321:role/RoleB';
```

L'exemple suivant accorde le privilège ASMEROLE à l'utilisateur `reg_user1` pour le rôle IAM `Redshift-Exfunc` pour créer des fonctions externes.

```
grant assumerole on 'arn:aws:iam::123456789012:role/Redshift-Exfunc'
```

```
to reg_user1 for external function;
```

L'exemple suivant accorde le privilège ASSUMEROLE à l'utilisateur `reg_user1` pour le rôle IAM `Redshift-model` pour créer des modèles de machine learning.

```
grant assumerole on 'arn:aws:iam::123456789012:role/Redshift-ML'
to reg_user1 for create model;
```

Exemples d'octroi de privilèges ROLE

L'exemple suivant accorde le rôle sample_role1 à l'utilisateur user1.

```
CREATE ROLE sample_role1;
GRANT ROLE sample_role1 TO user1;
```

L'exemple suivant accorde le rôle sample_role1 à l'utilisation user1 avec la clause WITH ADMIN OPTION, définit la séance actuelle pour user1, et user1 accorde le rôle sample_role1 à l'utilisateur user2.

```
GRANT ROLE sample_role1 TO user1 WITH ADMIN OPTION;
SET SESSION AUTHORIZATION user1;
GRANT ROLE sample_role1 TO user2;
```

L'exemple suivant accorde le rôle sample_role1 au rôle sample_role2.

```
GRANT ROLE sample_role1 TO ROLE sample_role2;
```

L'exemple suivant accorde le rôle sample_role2 au rôle sample_role3 et au rôle sample_role4. Ensuite, il tente d'accorder le rôle sample_role3 au rôle sample_role1.

```
GRANT ROLE sample_role2 TO ROLE sample_role3;
GRANT ROLE sample_role3 TO ROLE sample_role2;
ERROR: cannot grant this role, a circular dependency was detected between these roles
```

L'exemple suivant accorde les privilèges système CREATE USER au rôle sample_role1.

```
GRANT CREATE USER TO ROLE sample_role1;
```

L'exemple suivant accorde le rôle défini par le système `sys:dba` à l'utilisateur user1.

```
GRANT ROLE sys:dba TO user1;
```

L'exemple suivant tente d'accorder le rôle sample_role3 dans une dépendance circulaire au rôle sample_role2.

```
CREATE ROLE sample_role3;
GRANT ROLE sample_role2 TO ROLE sample_role3;
GRANT ROLE sample_role3 TO ROLE sample_role2; -- fail
ERROR:  cannot grant this role, a circular dependency was detected between these roles
```

INSERT

Rubriques

- Syntaxe
- Paramètres
- Notes d'utilisation
- Exemples INSERT

Insère de nouvelles lignes dans une table. Vous pouvez insérer une seule ligne avec la syntaxe VALUES, plusieurs lignes avec la syntaxe VALUES, ou une ou plusieurs lignes définies par les résultats d'une requête (INSERT INTO...SELECT).

> **Note**
>
> Nous vous encourageons vivement à utiliser la commande COPY pour charger de grandes quantités de données. La lenteur liée à l'utilisation d'instructions INSERT pour remplir une table peut être prohibitive. Sinon, si vos données existent déjà dans d'autres tables de bases de données Amazon Redshift, utilisez INSERT INTO SELECT ou CREATE TABLE AS pour améliorer les performances. Pour plus d'informations sur l'utilisation de la commande COPY pour charger les tables, consultez Chargement de données dans Amazon Redshift.

> **Note**
>
> La taille maximale d'une instruction SQL est de 16 Mo.

Syntaxe

```
INSERT INTO table_name [ ( column [, ...] ) ]
{DEFAULT VALUES |
VALUES ( { expression | DEFAULT } [, ...] )
[, ( { expression | DEFAULT } [, ...] )
[, ...] ] |
query }
```

Paramètres

table_name

Table temporaire ou permanente. Seul le propriétaire de la table ou un utilisateur avec le privilège INSERT sur la table peut insérer des lignes. Si vous utilisez la clause query pour insérer des lignes, vous devez avoir le privilège SELECT sur les tables nommées de la requête.

> **Note**
>
> Utilisez INSERT (table externe) pour insérer les résultats d'une requête SELECT dans des tables existantes du catalogue externe. Pour plus d'informations, consultez INSERT (table externe).

column

Vous pouvez insérer des valeurs dans une ou plusieurs colonnes de la table. Vous pouvez afficher les noms de colonne cible dans n'importe quel ordre. Si vous ne spécifiez pas une liste de colonnes, les valeurs à insérer doivent correspondre aux Colonnes de la table, dans l'ordre dans lequel elles ont été déclarés dans l'instruction CREATE TABLE. Si le nombre de valeurs à insérer est inférieur au nombre de colonnes de la table, les n premières colonnes sont chargées.

La valeur par défaut déclaré ou une valeur null est chargée dans l'une des colonnes qui n'est pas répertoriée (implicitement ou explicitement) dans l'instruction INSERT.

DEFAULT VALUES

Si des valeurs par défaut ont été attribuées aux colonnes de la table lorsque la table a été créée, utilisez ces mots-clés pour insérer une ligne qui se compose entièrement de valeurs par défaut. Si aucun des colonnes n'a de valeurs par défaut, des valeurs null sont insérées dans ces colonnes. Si aucune des colonnes n'est déclarée NOT NULL, l'instruction INSERT renvoie une erreur.

VALUES

Utilisez ce mot-clé pour insérer une ou plusieurs lignes, chaque ligne consistant en une ou plusieurs valeurs. La liste VALUES de chaque ligne doit être alignée avec la liste des colonnes. Pour insérer plusieurs lignes, utilisez une virgule de séparation entre chaque liste d'expressions. Ne répétez pas le mot-clé VALUES. Toutes les listes VALUES d'une instruction INSERT à plusieurs lignes doivent contenir le même nombre de valeurs.

expression

Une expression ou valeur unique analysée comme valeur unique. Chaque valeur doit être compatible avec le type de données de la colonne où elle est insérée. Si possible, une valeur dont le type de données ne correspond pas au type de données déclaré de la colonne est automatiquement convertie en un type de données compatible. Par exemple :

- Une valeur décimale `1.1` est insérée dans une colonne INT en tant que `1`.
- Une valeur décimale `100.8976` est insérée dans une colonne DEC(5,2) en tant que `100.90`.

Vous pouvez convertir explicitement une valeur en un type de données compatible en incluant la syntaxe de cast de type dans l'expression. Par exemple, si la colonne COL1 de la table T1 est une colonne CHAR (3) :

```
insert into t1(col1) values('Incomplete'::char(3));
```

Cette instruction insère la valeur `Inc` dans la colonne.

Pour une instruction INSERT VALUES à une seule valeur, vous pouvez utiliser une sous-requête scalaire comme expression. Le résultat de la sous-requête est inséré dans la colonne appropriée.

> **Note**
>
> Les sous-requêtes ne sont pas prises en charge comme expressions pour les instructions INSERT VALUES à plusieurs lignes.

DEFAULT

Utilisez ce mot-clé pour insérer la valeur par défaut d'une colonne, comme défini lors de la création de la table. S'il n'existe aucune valeur par défaut pour une colonne, une valeur null est insérée. Vous ne pouvez pas insérer une valeur par défaut dans une colonne ayant une contrainte

NOT NULL si cette colonne n'a pas une valeur par défaut explicite qui lui est attribuée dans l'instruction CREATE TABLE.

query

Insérez une ou plusieurs lignes dans la table en définissant une requête. Toutes les lignes que la requête produit sont insérées dans la table. La requête doit renvoyer une liste de colonnes compatible avec les colonnes de la table, mais les noms de colonnes n'ont pas à correspondre.

Notes d'utilisation

> **Note**
>
> Nous vous encourageons vivement à utiliser la commande COPY pour charger de grandes quantités de données. La lenteur liée à l'utilisation d'instructions INSERT pour remplir une table peut être prohibitive. Sinon, si vos données existent déjà dans d'autres tables de bases de données Amazon Redshift, utilisez INSERT INTO SELECT ou CREATE TABLE AS pour améliorer les performances. Pour plus d'informations sur l'utilisation de la commande COPY pour charger les tables, consultez Chargement de données dans Amazon Redshift.

Le format de données pour les valeurs insérées doit correspondre au format de données spécifié par la définition de CREATE TABLE.

Après l'insertion d'un grand nombre de nouvelles lignes dans une table :

- Exécuter une opération VACUUM sur la table pour récupérer de l'espace de stockage et retrier les lignes.
- Analysez la table pour mettre à jour les statistiques pour le planificateur de requête.

Lorsque les valeurs sont insérées dans des colonnes DECIMAL et qu'elles dépassent l'échelle spécifiée, les valeurs chargées sont arrondies le cas échéant. Par exemple, si la valeur `20.259` est insérée dans une colonne DECIMAL(8,2), la valeur est stockée sous la forme `20.26`.

Vous pouvez insérer dans une colonne GENERATED BY DEFAULT AS IDENTITY. Vous pouvez mettre à jour des colonnes définies comme GENERATED BY DEFAULT AS IDENTITY avec des valeurs que vous fournissez. Pour plus d'informations, consultez GENERATED BY DEFAULT AS IDENTITY.

Exemples INSERT

La table CATEGORY de la base de données TICKIT contient les lignes suivantes :

```
 catid | catgroup |  catname  |                 catdesc
-------+----------+-----------+--------------------------------------------
     1 | Sports   | MLB       | Major League Baseball
     2 | Sports   | NHL       | National Hockey League
     3 | Sports   | NFL       | National Football League
     4 | Sports   | NBA       | National Basketball Association
     5 | Sports   | MLS       | Major League Soccer
     6 | Shows    | Musicals  | Musical theatre
     7 | Shows    | Plays     | All non-musical theatre
     8 | Shows    | Opera     | All opera and light opera
     9 | Concerts | Pop       | All rock and pop music concerts
    10 | Concerts | Jazz      | All jazz singers and bands
    11 | Concerts | Classical | All symphony, concerto, and choir concerts
(11 rows)
```

Créez une table CATEGORY_STAGE avec un schéma similaire à la table CATEGORY, mais définissez les valeurs par défaut pour les colonnes :

```
create table category_stage
(catid smallint default 0,
catgroup varchar(10) default 'General',
catname varchar(10) default 'General',
catdesc varchar(50) default 'General');
```

L'instruction INSERT suivante sélectionne toutes les lignes de la table CATEGORY et les insère dans la table CATEGORY_STAGE.

```
insert into category_stage
(select * from category);
```

Les parenthèses autour de la requête sont facultatives.

Cette commande insère une nouvelle ligne dans la table CATEGORY_STAGE avec une valeur spécifiée pour chaque colonne dans l'ordre :

```
insert into category_stage values
```

```
(12, 'Concerts', 'Comedy', 'All stand-up comedy performances');
```

Vous pouvez également insérer une nouvelle ligne qui associe des valeurs spécifiques et des valeurs par défaut :

```
insert into category_stage values
(13, 'Concerts', 'Other', default);
```

Exécutez la requête suivante pour renvoyer les lignes insérées :

```
select * from category_stage
where catid in(12,13) order by 1;

 catid | catgroup | catname |             catdesc
-------+----------+---------+----------------------------------
    12 | Concerts | Comedy  | All stand-up comedy performances
    13 | Concerts | Other   | General
(2 rows)
```

Les exemples suivants illustrent quelques instructions INSERT VALUES à plusieurs lignes. Le premier exemple insère les valeurs spécifiques CATID pour deux lignes et les valeurs par défaut pour les autres colonnes des deux lignes.

```
insert into category_stage values
(14, default, default, default),
(15, default, default, default);

select * from category_stage where catid in(14,15) order by 1;
 catid | catgroup | catname | catdesc
-------+----------+---------+---------
    14 | General  | General | General
    15 | General  | General | General
(2 rows)
```

L’exemple suivant insère trois lignes avec différentes combinaisons de valeurs spécifiques et de valeurs par défaut :

```
insert into category_stage values
(default, default, default, default),
(20, default, 'Country', default),
(21, 'Concerts', 'Rock', default);
```

```
select * from category_stage where catid in(0,20,21) order by 1;
 catid | catgroup | catname | catdesc
-------+----------+---------+---------
     0 | General  | General | General
    20 | General  | Country | General
    21 | Concerts | Rock    | General
(3 rows)
```

Le premier ensemble de VALUES de cet exemple produit les mêmes résultats que la spécification de DEFAULT VALUES pour une instruction INSERT à ligne unique.

Les exemples suivants illustrent le comportement INSERT lorsqu'une table possède une colonne IDENTITY. D'abord, créez une nouvelle version de la table CATEGORY, puis insérez-y des lignes de CATEGORY :

```
create table category_ident
(catid int identity not null,
catgroup varchar(10) default 'General',
catname varchar(10) default 'General',
catdesc varchar(50) default 'General');


insert into category_ident(catgroup,catname,catdesc)
select catgroup,catname,catdesc from category;
```

Notez que vous ne pouvez pas insérer des valeurs entières spécifiques dans la colonne CATID IDENTITY. Les valeurs de colonne IDENTITY sont générées automatiquement.

L'exemple suivant montre que les sous-requêtes ne peuvent pas être utilisées comme expressions dans les instructions INSERT VALUES à plusieurs lignes :

```
insert into category(catid) values
((select max(catid)+1 from category)),
((select max(catid)+2 from category));

ERROR: can't use subqueries in multi-row VALUES
```

L'exemple suivant montre une insertion dans une table temporaire remplie avec les données de la table `venue` à l'aide de la clause `WITH SELECT`. Pour plus d'informations sur la table `venue`, consultez Exemple de base de données.

Commencez par créer la table temporaire #venuetemp.

```
CREATE TABLE #venuetemp AS SELECT * FROM venue;
```

Répertoriez les lignes de la table #venuetemp.

```
SELECT * FROM #venuetemp ORDER BY venueid;

venueid | venuename                | venuecity  | venuestate| venueseats
--------+--------------------------+------------+-----------+------------
1       Toyota Park                Bridgeview   IL          0
2       Columbus Crew Stadium      Columbus     OH          0
3       RFK Stadium                Washington   DC          0
4       CommunityAmerica Ballpark  Kansas City  KS          0
5       Gillette Stadium           Foxborough   MA          68756
...
```

Insérez 10 lignes dupliquées dans la table #venuetemp à l'aide de la clause WITH SELECT.

```
INSERT INTO #venuetemp (WITH venuecopy AS (SELECT * FROM venue) SELECT * FROM venuecopy
 ORDER BY 1 LIMIT 10);
```

Répertoriez les lignes de la table #venuetemp.

```
SELECT * FROM #venuetemp ORDER BY venueid;

venueid | venuename                | venuecity  | venuestate| venueseats
--------+--------------------------+------------+-----------+------------
1       Toyota Park                Bridgeview   IL          0
1       Toyota Park                Bridgeview   IL          0
2       Columbus Crew Stadium      Columbus     OH          0
2       Columbus Crew Stadium      Columbus     OH          0
3       RFK Stadium                Washington   DC          0
3       RFK Stadium                Washington   DC          0
4       CommunityAmerica Ballpark  Kansas City  KS          0
4       CommunityAmerica Ballpark  Kansas City  KS          0
5       Gillette Stadium           Foxborough   MA          68756
5       Gillette Stadium           Foxborough   MA          68756
...
```

INSERT (table externe)

Insère les résultats d'une requête SELECT dans des tables externes existantes d'un catalogue externe, telles que for AWS Glue AWS Lake Formation, ou une métastore Apache Hive. Utilisez le même rôle AWS Identity and Access Management (IAM) que celui utilisé pour la commande CREATE EXTERNAL SCHEMA pour interagir avec les catalogues externes et Amazon S3.

Pour les tables non partitionnées, la commande INSERT (table externe) écrit les données à l'emplacement Amazon S3 défini dans la table, en fonction des propriétés de la table et du format de fichier spécifiés.

Pour les tables partitionnées, INSERT (table externe) écrit les données à l'emplacement Amazon S3 en fonction de la clé de partition spécifiée dans la table. Il enregistre également automatiquement les nouvelles partitions dans le catalogue externe une fois l'opération INSERT terminée.

Vous ne pouvez pas exécuter INSERT (table externe) dans un bloc de transactions (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.

Syntaxe

```
INSERT INTO external_schema.table_name
{ select_statement }
```

Paramètres

schéma_externe.nom_table

Nom d'un schéma externe existant et d'une table externe cible pour l'insertion.

select_statement

Instruction qui insère une ou plusieurs lignes dans la table externe en définissant une requête. Toutes les lignes produites par la requête sont écrites dans Amazon S3 au format texte ou Parquet, en fonction de la définition de la table. La requête doit renvoyer une liste de colonnes compatible avec les types de données de colonne de la table externe. Cependant, les noms des colonnes ne doivent pas obligatoirement correspondre.

Notes d'utilisation

Le nombre de colonnes de la requête SELECT doit être identique à la somme des colonnes de données et des colonnes de partition. L'emplacement et le type de données de chaque colonne

de données doivent correspondre à ceux de la table externe. L'emplacement des colonnes de partition doit se trouver à la fin de la requête SELECT, dans le même ordre que celui utilisé pour les définir dans la commande CREATE EXTERNAL TABLE. Les noms des colonnes ne doivent pas obligatoirement correspondre.

Dans certains cas, vous pouvez exécuter la commande INSERT (table externe) sur un catalogue de données AWS Glue ou un métastore Hive. Dans le cas de AWS Glue, le rôle IAM utilisé pour créer le schéma externe doit disposer à la fois d'autorisations de lecture et d'écriture sur Amazon S3 et AWS Glue. Si vous utilisez un AWS Lake Formation catalogue, ce rôle IAM devient propriétaire de la nouvelle table Lake Formation. Ce rôle IAM doit au moins disposer des autorisations suivantes :

- Autorisation SELECT, INSERT, UPDATE sur la table externe
- Autorisation d'emplacement des données sur le chemin d'accès Amazon S3 de la table externe

Pour vous assurer que les noms de fichiers sont uniques, Amazon Redshift utilise le format suivant pour le nom de chaque fichier téléchargé dans Amazon S3 par défaut.

*<date>*_*<time>*_*<microseconds>*_*<query_id>*_*<slice-number>*_part_*<part-number>*.*<format>*.

Par exemple : `20200303_004509_810669_1007_0001_part_00.parquet`.

Tenez compte des éléments suivants lors de l'exécution de la commande INSERT (table externe) :

- Les tables externes qui ont un format autre que PARQUET ou TEXTFILE ne sont pas prises en charge.
- Cette commande prend en charge les propriétés de table existantes telles que 'write.parallel', 'write.maxfilesize.mb', 'compression_type' et 'serialization.null.format'. Pour mettre à jour ces valeurs, exécutez la commande ALTER TABLE SET TABLE PROPERTIES.
- La propriété de table 'numRows' est automatiquement mise à jour vers la fin de l'opération INSERT. La propriété de la table doit déjà être définie ou ajoutée à la table si elle n'a pas été créée par l'opération CREATE EXTERNAL TABLE AS.
- La clause LIMIT n'est pas prise en charge dans la requête SELECT externe. A la place, utilisez une clause LIMIT imbriquée.
- Vous pouvez utiliser la table STL_UNLOAD_LOG pour suivre les fichiers qui ont été écrits sur Amazon S3 par chaque opération INSERT (table externe).

- Amazon Redshift prend en charge uniquement le chiffrement standard Amazon S3 pour INSERT (table externe).

Exemples d'opération INSERT (table externe)

L'exemple suivant insère les résultats de l'instruction SELECT dans la table externe.

```
INSERT INTO spectrum.lineitem
SELECT * FROM local_lineitem;
```

L'exemple suivant insère les résultats de l'instruction SELECT dans une table externe partitionnée à l'aide d'un partitionnement statique. Les colonnes de partition sont codées de manière irréversible dans l'instruction SELECT. Les colonnes de partition doivent se trouver à la fin de la requête.

```
INSERT INTO spectrum.customer
SELECT name, age, gender, 'May', 28 FROM local_customer;
```

L'exemple suivant insère les résultats de l'instruction SELECT dans une table externe partitionnée à l'aide d'un partitionnement dynamique. Les colonnes de partition ne sont pas codées de manière irréversible. Les données sont automatiquement ajoutées aux dossiers de partition existants ou aux nouveaux dossiers si une nouvelle partition est ajoutée.

```
INSERT INTO spectrum.customer
SELECT name, age, gender, month, day FROM local_customer;
```

LOCK

Limite l'accès à une table de base de données. Cette commande n'est significative que lorsqu'elle est exécutée à l'intérieur d'un bloc de transaction.

La commande LOCK obtient un verrou de niveau table en mode « ACCESS EXCLUSIVE », attendant si nécessaire que des verrous en conflit soient libérés. Un tel verrou explicite d'une table peut entraîner une attente des lectures et des écritures sur la table lorsqu'elles sont tentées à partir d'autres transactions ou séances. Un verrou de table explicite créé par un utilisateur empêche temporairement un autre utilisateur de sélectionner les données de cette table ou de les y charger. Le verrou est libéré lorsque la transaction qui contient la commande LOCK est terminée.

Les verrous de table moins restrictifs sont acquis implicitement par les commandes qui font référence aux tables, telles que les opérations d'écriture. Par exemple, si un utilisateur tente de lire les données

d'une table pendant qu'un autre utilisateur met à jour la table, les données qui sont lues constituent un instantané des données déjà validées. (Dans certains cas, les requêtes sont arrêtées si elles violent les règles d'isolement sérialisable.) Consultez Gestion des opérations d'écriture simultanées.

Certaines opérations DDL, telles que DROP TABLE et TRUNCATE, créent des verrous exclusifs. Ces opérations empêchent les lectures de données.

Si un conflit de verrou se produit, Amazon Redshift affiche un message d'erreur pour avertir l'utilisateur qui a démarré la transaction en conflit. La transaction qui a reçu le conflit de verrou est arrêtée. Chaque fois qu'un conflit de verrou se produit, Amazon Redshift écrit une entrée dans la table STL_TR_CONFLICT.

Syntaxe

```
LOCK [ TABLE ] table_name [, ...]
```

Paramètres

TABLE

Mot-clé facultatif.

table_name

Nom de la table à verrouiller. Vous pouvez verrouiller plusieurs tables en utilisant une liste de noms de table séparés par des virgules. Vous ne pouvez pas verrouiller les vues.

exemple

```
begin;

lock event, sales;

...
```

MERGE

Fusionne de manière conditionnelle les lignes d'une table source dans une table cible. Traditionnellement, cela ne peut être réalisé qu'en utilisant plusieurs instructions d'insertion, de mise

à jour ou de suppression séparément. Pour plus d'informations sur les opérations que MERGE vous permet de combiner, consultez UPDATE, DELETE et INSERT.

Syntaxe

```
MERGE INTO target_table
USING source_table [ [ AS ] alias ]
ON match_condition
[ WHEN MATCHED THEN { UPDATE SET col_name = { expr } [,...] | DELETE }
WHEN NOT MATCHED THEN INSERT [ ( col_name [,...] ) ] VALUES ( { expr } [, ...] ) |
REMOVE DUPLICATES ]
```

Paramètres

target_table

Table temporaire ou permanente dans laquelle l'instruction MERGE est fusionnée.

source_table

La table temporaire ou permanente fournissant les lignes à fusionner dans target_table. source_table peut également être une table Spectrum.

alias

Nom alternatif temporaire pour source_table.

Ce paramètre est facultatif. L'alias précédent avec AS est également facultatif.

match_condition

Spécifie des prédicats égaux entre la colonne de la table source et la colonne de la table cible qui sont utilisés pour déterminer si les lignes de source_table peuvent correspondre aux lignes de target_table. Si la condition est remplie, MERGE exécute matched_clause pour cette ligne. Sinon, MERGE exécute not_matched_clause pour cette ligne.

WHEN MATCHED

Spécifie l'action à exécuter lorsque la condition de correspondance entre une ligne source et une ligne cible est évaluée sur True. Vous pouvez spécifier une action UPDATE ou une action DELETE.

UPDATE

Met à jour la ligne correspondante dans target_table. Seules les valeurs dans col_name que vous spécifiez sont mises à jour.

DELETE

Supprime la ligne correspondante dans target_table.

WHEN NOT MATCHED

Spécifie l'action à exécuter lorsque la condition de correspondance est évaluée sur False ou Unknown. Vous pouvez uniquement spécifier l'action d'insertion INSERT pour cette clause.

INSERT

Insère dans target_table des lignes de source_table qui ne correspondent à aucune ligne de target_table, selon match_condition. col_name cible peut être répertorié dans n'importe quel ordre. Si vous ne fournissez aucune valeur col_name, l'ordre par défaut correspond à l'ordre déclaré de toutes les colonnes de la table.

col_name

Un ou plusieurs noms de colonnes que vous voulez modifier. N'incluez pas le nom de la table quand vous spécifiez la colonne cible.

expr

Expression définissant la nouvelle valeur pour col_name.

REMOVE DUPLICATES

Spécifie que la commande MERGE s'exécute en mode simplifié. Le mode simplifié a les exigences suivantes :

- target_table et source_table doivent avoir le même nombre de colonnes et les mêmes types de colonnes compatibles.
- Omettez la clause WHEN et les clauses UPDATE et INSERT de votre commande MERGE.
- Utilisez la clause REMOVE DUPLICATES dans votre commande MERGE.

En mode simplifié, MERGE effectue les opérations suivantes :

- Les lignes dans target_table qui ont une correspondance dans source_table sont mises à jour pour correspondre aux valeurs dans source_table.

- Les lignes dans source_table qui n'ont pas de correspondance dans target_table sont insérées dans target_table.
- Lorsque plusieurs lignes dans target_table correspondent à la même ligne dans source_table, les lignes dupliquées sont supprimées. Amazon Redshift conserve une ligne et la met à jour. Les lignes dupliquées qui ne correspondent pas à une ligne dans source_table restent inchangées.

L'utilisation de REMOVE DUPLICATES donne de meilleures performances que l'utilisation de WHEN MATCHED et WHEN NOT MATCHED. Nous vous recommandons d'utiliser REMOVE DUPLICATES si target_table et source_table sont compatibles et si vous n'avez pas besoin de conserver les lignes dupliquées dans target_table.

Notes d'utilisation

- Pour exécuter des instructions MERGE, vous devez être propriétaire à la fois de source_table et de target_table, ou disposer de l'autorisation SELECT pour ces tables. En outre, vous devez disposer des autorisations UPDATE, DELETE et INSERT pour target_table en fonction des opérations incluses dans votre instruction MERGE.
- target_table ne peut pas être une table système, une table de catalogue ou une table externe.
- source_table et target_table ne peuvent pas être la même table.
- Vous ne pouvez pas utiliser la clause WITH dans une instruction MERGE.
- Les lignes de la target_table ne peuvent pas correspondre à plusieurs lignes de la source_table.

Prenez l'exemple suivant :

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (1, 'Bob'), (2, 'John');
INSERT INTO source VALUES (1, 'Tony'), (1, 'Alice'), (3, 'Bill');

MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN UPDATE SET id = source.id, name = source.name
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);
ERROR: Found multiple matches to update the same tuple.

MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN DELETE
```

```
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);
ERROR: Found multiple matches to update the same tuple.
```

Dans les deux instructions MERGE, l'opération échoue parce qu'il y a plusieurs lignes dans la table `source` avec une valeur ID de 1.

- match_condition et expr ne peuvent pas référencer partiellement les colonnes de type SUPER. Par exemple, si votre objet de type SUPER est un tableau ou une structure, vous ne pouvez pas utiliser des éléments individuels de cette colonne pour match_condition ou expr, mais vous pouvez utiliser la colonne entière.

  Prenez l'exemple suivant :

```
CREATE TABLE IF NOT EXISTS target (key INT, value SUPER);
CREATE TABLE IF NOT EXISTS source (key INT, value SUPER);

INSERT INTO target VALUES (1, JSON_PARSE('{"key": 88}'));
INSERT INTO source VALUES (1, ARRAY(1, 'John')), (2, ARRAY(2, 'Bill'));

MERGE INTO target USING source ON target.key = source.key
WHEN matched THEN UPDATE SET value = source.value[0]
WHEN NOT matched THEN INSERT VALUES (source.key, source.value[0]);
ERROR: Partial reference of SUPER column is not supported in MERGE statement.
```

  Pour plus d'informations sur le type SUPER, consultez Type SUPER.

- Si source_table est volumineuse, définir les colonnes de jointure de target_table et de source_table comme clés de distribution peut améliorer les performances.
- Pour utiliser la clause REMOVE DUPLICATES, vous devez disposer des autorisations SELECT, INSERT et DELETE pour target_table.
- source_table peut être une vue ou une sous-requête. Voici un exemple d'instruction MERGE dans laquelle source_table est une sous-requête qui supprime les lignes dupliquées.

```
MERGE INTO target
USING (SELECT id, name FROM source GROUP BY 1, 2) as my_source
ON target.id = my_source.id
WHEN MATCHED THEN UPDATE SET id = my_source.id, name = my_source.name
WHEN NOT MATCHED THEN INSERT VALUES (my_source.id, my_source.name);
```

- La cible ne peut pas être la source de données d'une sous-requête de la même instruction MERGE. Par exemple, la commande SQL suivante renvoie une erreur du type ERROR :

L'instruction Source view/subquery in Merge ne peut pas référencer la table cible. car la sous-requête fait référence `target` au lieu de. `source`

```
MERGE INTO target
USING (SELECT id, name FROM target GROUP BY 1, 2) as my_source
ON target.id = my_source.id
WHEN MATCHED THEN UPDATE SET id = my_source.id, name = my_source.name
WHEN NOT MATCHED THEN INSERT VALUES (my_source.id, my_source.name);
```

Exemples

L'exemple suivant crée deux tables, puis exécute une opération MERGE sur celles-ci qui met à jour les lignes correspondantes dans la table cible et insère des lignes qui ne correspondent pas. Elle insère ensuite une autre valeur dans la table source et exécute une autre opération MERGE, qui supprime cette fois les lignes correspondantes et insère la nouvelle ligne de la table source.

Créez et remplissez d'abord les tables source et cible.

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (101, 'Bob'), (102, 'John'), (103, 'Susan');
INSERT INTO source VALUES (102, 'Tony'), (103, 'Alice'), (104, 'Bill');

SELECT * FROM target;
 id  |    name
-----+------------
 101 | Bob
 102 | John
 103 | Susan
(3 rows)

SELECT * FROM source;
 id  |    name
-----+------------
 102 | Tony
 103 | Alice
 104 | Bill
(3 rows)
```

Fusionnez ensuite la table source dans la table cible, ce qui met à jour la table cible avec les lignes correspondantes et insère les lignes de la table source qui n'ont aucune correspondance.

```
MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN UPDATE SET id = source.id, name = source.name
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);

SELECT * FROM target;
 id  |    name
-----+------------
 101 | Bob
 102 | Tony
 103 | Alice
 104 | Bill
(4 rows)
```

Notez que les lignes dont les valeurs d'id sont 102 et 103 sont mises à jour pour correspondre aux valeurs de nom de la table cible. En outre, une nouvelle ligne avec une valeur d'id 104 et une valeur de nom Bill est insérée dans la table cible.

Ensuite, insérez une nouvelle ligne dans la table source.

```
INSERT INTO source VALUES (105, 'David');

SELECT * FROM source;
 id  |    name
-----+------------
 102 | Tony
 103 | Alice
 104 | Bill
 105 | David
(4 rows)
```

Enfin, exécutez une opération de fusion en supprimant les lignes correspondantes dans la table cible et en insérant les lignes qui ne correspondent pas.

```
MERGE INTO target USING source ON target.id = source.id
WHEN MATCHED THEN DELETE
WHEN NOT MATCHED THEN INSERT VALUES (source.id, source.name);

SELECT * FROM target;
```

```
 id  |    name
-----+------------
 101 | Bob
 105 | David
(2 rows)
```

Les lignes avec les valeurs d'id 102, 103 et 104 sont supprimées de la table cible, et une nouvelle ligne avec une valeur d'id de 105 et la valeur de nom David est insérée dans la table cible.

L'exemple suivant montre la syntaxe simplifiée d'une commande MERGE qui utilise la clause REMOVE DUPLICATES.

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (30, 'Tony'), (11, 'Alice'), (23, 'Bill');
INSERT INTO source VALUES (23, 'David'), (22, 'Clarence');

MERGE INTO target USING source ON target.id = source.id REMOVE DUPLICATES;

SELECT * FROM target;
id | name
---+------------
30 | Tony
11 | Alice
23 | David
22 | Clarence
(4 rows)
```

L'exemple suivant montre la syntaxe simplifiée d'une commande MERGE qui utilise la clause REMOVE DUPLICATES, qui supprime les lignes dupliquées de target_table si elles ont des lignes correspondantes dans source_table.

```
CREATE TABLE target (id INT, name CHAR(10));
CREATE TABLE source (id INT, name CHAR(10));

INSERT INTO target VALUES (30, 'Tony'), (30, 'Daisy'), (11, 'Alice'), (23, 'Bill'),
 (23, 'Nikki');
INSERT INTO source VALUES (23, 'David'), (22, 'Clarence');

MERGE INTO target USING source ON target.id = source.id REMOVE DUPLICATES;
```

```
SELECT * FROM target;
id | name
---+------------
30 | Tony
30 | Daisy
11 | Alice
23 | David
22 | Clarence
(5 rows)
```

Après l'exécution de MERGE, il n'y a qu'une seule ligne avec une valeur d'ID 23 dans target_table. Comme aucune ligne n'avait la valeur d'ID 30 dans source_table, les deux lignes dupliquées avec des valeurs d'ID 30 restent dans target_table.

Consultez aussi

INSERT, UPDATE, DELETE

PREPARE

Prépare une instruction pour l'exécution.

PREPARE crée une instruction préparée. Lorsque l'instruction PREPARE est exécutée, l'instruction spécifiée (SELECT, INSERT, UPDATE ou DELETE) est analysée, réécrite et planifiée. Quand une commande EXECUTE est ensuite émise pour l'instruction préparée, Amazon Redshift peut le cas échéant réviser le plan d'exécution de la requête (pour améliorer les performances en fonction des valeurs de paramètre spécifiées) avant l'exécution de l'instruction préparée.

Syntaxe

```
PREPARE plan_name [ (datatype [, ...] ) ] AS statement
```

Paramètres

nom_plan

Nom arbitraire donné à cette instruction préparée particulière. Il doit être unique au sein d'une même séance et est ensuite utilisé pour exécuter ou libérer une instruction préalablement préparée.

datatype

Type de données d'un paramètre de l'instruction préparée. Pour faire référence aux paramètres de l'instruction préparée elle-même, utilisez 1$, 2$, etc., jusqu'à un maximum de 32 767$.

instruction

Toute instruction SELECT, INSERT, UPDATE ou DELETE.

Notes d'utilisation

Les instructions préparées peuvent accepter des paramètres : les valeurs qui sont remplacées dans l'instruction lors de l'exécution. Pour inclure des paramètres dans une instruction préparée, fournissez une liste de types de données dans l'instruction PREPARE et, dans l'instruction à préparer elle-même, faites référence aux paramètres de position à l'aide de la notation $1, $2, etc. Le nombre maximal de paramètres est 32 767. Lors de l'exécution de l'instruction, spécifiez les valeurs réelles de ces paramètres dans l'instruction EXECUTE. Pour en savoir plus, consultez EXECUTE.

Les instructions préparées ne sont valables que sur la durée de la séance en cours. Lorsque la séance se termine, comme l'instruction préparée est écartée, elle devra être recréée avant d'être utilisée à nouveau. Cela signifie également qu'une même instruction préparée ne peut pas être utilisée simultanément par plusieurs clients de base de données ; néanmoins, chaque client peut créer sa propre instruction à utiliser. L'instruction préparée peut être supprimée manuellement à l'aide de la commande DEALLOCATE.

Les instructions préparées ont l'avantage d'offrir les meilleures performances lorsqu'une même séance est utilisée pour exécuter un grand nombre d'instructions similaires. Comme mentionné, pour chaque nouvelle exécution d'une instruction préparée, Amazon Redshift peut réviser le plan d'exécution afin d'améliorer les performances en fonction des valeurs de paramètre spécifiées. Pour examiner le plan d'exécution de requête qu'Amazon Redshift a choisi pour des instructions EXECUTE spécifiques, utilisez la commande EXPLAIN.

Pour plus d'informations sur la planification des requêtes et les statistiques recueillies par Amazon Redshift pour l'optimisation des requêtes, consultez la commande ANALYSE.

Exemples

Créez une table temporaire, préparez l'instruction INSERT, puis exécutez-la :

```
DROP TABLE IF EXISTS prep1;
```

```
CREATE TABLE prep1 (c1 int, c2 char(20));
PREPARE prep_insert_plan (int, char)
AS insert into prep1 values ($1, $2);
EXECUTE prep_insert_plan (1, 'one');
EXECUTE prep_insert_plan (2, 'two');
EXECUTE prep_insert_plan (3, 'three');
DEALLOCATE prep_insert_plan;
```

Préparez une instruction SELECT, puis exécutez-la :

```
PREPARE prep_select_plan (int)
AS select * from prep1 where c1 = $1;
EXECUTE prep_select_plan (2);
EXECUTE prep_select_plan (3);
DEALLOCATE prep_select_plan;
```

Consultez aussi

DEALLOCATE, EXECUTE

REFRESH MATERIALIZED VIEW

Actualise une vue matérialisée.

Lorsque vous créez une vue matérialisée, son contenu reflète l'état de la ou des tables de base de données sous-jacentes à ce moment-là. Les données de la vue matérialisée restent inchangées, même si les applications modifient les données dans les tables sous-jacentes.

Pour mettre à jour les données de la vue matérialisée, vous pouvez utiliser l'instruction `REFRESH MATERIALIZED VIEW` à tout moment. Lorsque vous utilisez cette instruction, Amazon Redshift identifie les modifications qui ont eu lieu dans la ou les tables de base, puis applique ces modifications à la vue matérialisée.

Pour plus d'informations sur les vues matérialisées, consultez Vues matérialisées dans Amazon Redshift.

Syntaxe

```
REFRESH MATERIALIZED VIEW mv_name
```

Paramètres

mv_name

Nom de la vue matérialisée à actualiser.

Notes d'utilisation

Seul le propriétaire d'une vue matérialisée peut effectuer une opération `REFRESH MATERIALIZED VIEW` sur cette vue matérialisée. En outre, le propriétaire doit disposer du privilège SELECT au niveau des tables de base sous-jacentes pour exécuter correctement `REFRESH MATERIALIZED VIEW`.

La commande `REFRESH MATERIALIZED VIEW` s'exécute en tant que transaction. La sémantique de transaction Amazon Redshift est respectée pour déterminer quelles données des tables de base sont accessibles à la commande `REFRESH` ou quand les modifications apportées par la commande `REFRESH` deviennent accessibles à d'autres transactions exécutées dans Amazon Redshift.

- Pour les vues matérialisées incrémentielles, `REFRESH MATERIALIZED VIEW` utilise uniquement les lignes de la table de base qui sont déjà validées. Par conséquent, si l'opération d'actualisation s'exécute après une instruction DML (Data Manipulation Language) dans la même transaction, les modifications de cette instruction DML ne sont pas visibles pour l'actualisation.
- Pour une actualisation complète d'une vue matérialisée, `REFRESH MATERIALIZED VIEW` voit toutes les lignes de table de base visibles par la transaction d'actualisation, selon la sémantique de transaction Amazon Redshift habituelle.
- Selon le type d'argument d'entrée, Amazon Redshift prend toujours en charge l'actualisation incrémentielle pour les vues matérialisées pour les fonctions suivantes avec des types d'arguments d'entrée spécifiques : DATE (horodatage), DATE_PART (date, heure, intervalle, heure-tz), DATE_TRUNC (horodatage, intervalle).
- L'actualisation incrémentielle est également prise en charge sur une vue matérialisée dont la table de base est une unité de partage des données.

Certaines opérations dans Amazon Redshift interagissent avec les vues matérialisées. Certaines de ces opérations peuvent forcer une opération `REFRESH MATERIALIZED VIEW` à recalculer complètement la vue matérialisée, même si la requête qui définit cette dernière utilise uniquement les fonctionnalités SQL éligibles à l'actualisation incrémentielle. Par exemple :

- Les opérations VACUUM en arrière-plan peuvent être bloquées si les vues matérialisées ne sont pas actualisées. Après une période de seuil définie en interne, l'exécution de l'opération VACUUM est autorisée. Lorsque cette opération VACUUM se produit, toutes les vues matérialisées dépendantes sont marquées pour le recalcul lors de l'actualisation suivante (même si elles sont incrémentielles). Pour obtenir des informations sur VACUUM, consultez VACUUM. Pour plus d'informations sur les événements et les changements d'état, consultez STL_MV_STATE.
- Certaines opérations effectuées par l'utilisateur sur les tables de base forcent le recalcul d'une vue matérialisée lors de l'exécution suivante d'une opération REFRESH. Des exemples d'opération de ce type sont une opération VACUUM appelée manuellement, un redimensionnement classique, une opération ALTER DISTKEY, une opération ALTER SORTKEY et une opération TRUNCATE. Dans certains cas, les opérations automatiques peuvent également entraîner le recalcul complet d'une vue matérialisée lors de la prochaine exécution d'une opération REFRESH. Par exemple, une opération de suppression automatique par aspiration peut entraîner un recalcul complet. Pour plus d'informations sur les événements et les changements d'état, consultez STL_MV_STATE.

Actualisation incrémentielle des vues matérialisées dans un partage de données

Amazon Redshift prend en charge l'actualisation automatique et incrémentielle des vues matérialisées dans un partage de données client lorsque les tables de base sont partagées. L'actualisation incrémentielle est une opération au cours de laquelle Amazon Redshift identifie les modifications apportées à la table de base ou aux tables après l'actualisation précédente et met à jour uniquement les enregistrements correspondants dans la vue matérialisée. Pour plus d'informations sur ce comportement, consultez CREATE MATERIALIZED VIEW.

Limites d'actualisation incrémentielle

Pour l'instant, Amazon Redshift ne prend pas en charge l'actualisation incrémentielle pour les vues matérialisées qui sont définies avec une requête utilisant l'un des éléments SQL suivants :

- OUTER JOIN (RIGHT, LEFT ou FULL).
- Opérations d'ensemble : UNION, INTERSECT, EXCEPT, MINUS.
- UNION ALL lorsqu'elle se produit dans une sous-requête et qu'une fonction d'agrégation ou une clause GROUP BY est présente dans la requête.
- Fonctions d'agrégation : MEDIAN, PERCENTILE_CONT, LISTAGG, STDDEV_SAMP, STDDEV_POP, APPROXIMATE COUNT, APPROXIMATE PERCENTILE et les fonctions d'agrégation bit par bit.

> **Note**
>
> Les fonctions d'agrégation COUNT, SUM, MIN, MAX et AVG sont prises en charge.

- Fonctions d'agrégation DISTINCT, telles que DISTINCT COUNT, DISTINCT SUM, etc.
- Fonctions de fenêtrage.
- Requête qui utilise des tables temporaires pour l'optimisation des requêtes, telle que l'optimisation des sous-expressions courantes.
- Sous-requêtes
- Tables externes référençant les formats suivants dans la requête qui définit la vue matérialisée.
  - Delta Lake
  - Hudi

  L'actualisation incrémentielle est prise en charge pour les vues matérialisées définies à l'aide de formats autres que ceux répertoriés ci-dessus. Pour de plus amples informations, veuillez consulter Vues matérialisées sur des tables de lacs de données externes dans Amazon Redshift Spectrum.
- Fonctions mutables, telles que les fonctions date-heure, RANOM et non-STABLE définies par l'utilisateur.
- Pour connaître les limites relatives à l'actualisation incrémentielle pour les intégrations sans ETL, consultez la section Considérations relatives à l'utilisation d'intégrations sans ETL avec Amazon Redshift.
- Accès aux tables depuis plusieurs bases de données.

Pour plus d'informations sur les limites de la vue matérialisée, y compris l'effet des opérations d'arrière-plan telles que VACUUM sur les opérations d'actualisation de la vue matérialisée, consultez Notes d'utilisation.

Exemples

L'exemple suivant actualise la vue matérialisée `tickets_mv`.

```
REFRESH MATERIALIZED VIEW tickets_mv;
```

RESET

Restaure la value d'un paramètre de configuration à sa valeur par défaut.

Vous pouvez réinitialiser un seul paramètre ou tous les paramètres à la fois. Pour définir un paramètre sur une valeur spécifique, utilisez la commande SET. Pour afficher la valeur actuelle d'un paramètre, utilisez la commande MONTRER.

Syntaxe

```
RESET { parameter_name | ALL }
```

L'instruction suivante définit la valeur d'une variable de contexte de session sur NULL.

```
RESET { variable_name | ALL }
```

Paramètres

nom_paramètre

Nom du paramètre à réinitialiser. Pour plus d'informations sur les paramètres, consultez Modification de la configuration du serveur.

ALL

Réinitialise tous les paramètres d'exécution, y compris toutes les variables de contexte de session.

variable

Le nom de la variable à réinitialiser. Si la valeur à RÉINITIALISER est une variable de contexte de session, Amazon Redshift la définit sur NULL.

Exemples

L'exemple suivant réinitialise le paramètre `query_group` à sa valeur par défaut :

```
reset query_group;
```

L'exemple suivant réinitialise tous les paramètres d'exécution à leurs valeurs par défaut.

```
reset all;
```

L'exemple suivant réinitialise la variable de contexte.

```
RESET app_context.user_id;
```

REVOKE

Supprime les autorisations d'accès d'un utilisateur ou d'un rôle, comme celles permettant de créer, de supprimer ou de mettre à jour des tables.

Vous ne pouvez accorder (GRANT) ou révoquer l'utilisation (REVOKE USAGE) sur un schéma externe à des utilisateurs et des rôles de base de données qu'en utilisant la syntaxe ON SCHEMA. Lorsque vous utilisez ON EXTERNAL SCHEMA avec AWS Lake Formation, vous pouvez uniquement ACCORDER et RÉVOQUER des autorisations pour un rôle AWS Identity and Access Management (IAM). Pour connaître la liste des autorisations, reportez-vous à la syntaxe.

Pour les procédures stockées, les autorisations USAGE ON LANGUAGE `plpgsql` sont accordées par défaut à PUBLIC. L'autorisation EXECUTE ON PROCEDURE est uniquement accordée au propriétaire et aux super-utilisateurs par défaut.

Spécifiez dans la commande REVOKE les autorisations que vous souhaitez supprimer. Pour accorder des autorisations, utilisez la commande GRANT.

Syntaxe

```
REVOKE [ GRANT OPTION FOR ]
{ { SELECT | INSERT | UPDATE | DELETE | DROP | REFERENCES | ALTER | TRUNCATE } [,...] |
 ALL [ PRIVILEGES ] }
ON { [ TABLE ] table_name [, ...] | ALL TABLES IN SCHEMA schema_name [, ...] }
FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
{ { CREATE | TEMPORARY | TEMP | ALTER } [,...] | ALL [ PRIVILEGES ] }
ON DATABASE db_name [, ...]
FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
{ { CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }
```

```
ON SCHEMA schema_name [, ...]
FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
EXECUTE
    ON FUNCTION function_name ( [ [ argname ] argtype [, ...] ] ) [, ...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
{ { EXECUTE } [,...] | ALL [ PRIVILEGES ] }
    ON PROCEDURE procedure_name ( [ [ argname ] argtype [, ...] ] ) [, ...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
USAGE
    ON LANGUAGE language_name [, ...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
[ RESTRICT ]

REVOKE [GRANT OPTION FOR]
{ { ALTER | DROP} [,...] | ALL [ PRIVILEGES ] }
    ON COPY JOB job_name [,...]
    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Révoquer les autorisations au niveau des colonnes pour les tables

Voici la syntaxe des autorisations de niveau colonne sur les tables et les vues Amazon Redshift.

```
REVOKE { { SELECT | UPDATE } ( column_name [, ...] ) [, ...] | ALL [ PRIVILEGES ]
 ( column_name [,...] ) }
     ON { [ TABLE ] table_name [, ...] }
     FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
     [ RESTRICT ]
```

Révoquer les autorisations ASSUMEROLE

La syntaxe suivante permet de révoquer l’autorisation ASSUMEROLE pour les utilisateurs et les groupes ayant un rôle spécifique.

```
REVOKE ASSUMEROLE
```

```
    ON { 'iam_role' [, ...]  | default | ALL }
    FROM { user_name | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
    FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL }
```

Révoquer les autorisations pour Redshift Spectrum pour Lake Formation

La syntaxe suivante s'applique à l'intégration de Redshift Spectrum à Lake Formation.

```
REVOKE [ GRANT OPTION FOR ]
{ SELECT | ALL [ PRIVILEGES ] } ( column_list )
    ON EXTERNAL TABLE schema_name.table_name
    FROM { IAM_ROLE iam_role } [, ...]

REVOKE [ GRANT OPTION FOR ]
{ { SELECT | ALTER | DROP | DELETE | INSERT }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL TABLE schema_name.table_name [, ...]
    FROM { { IAM_ROLE iam_role } [, ...] | PUBLIC }

REVOKE [ GRANT OPTION FOR ]
{ { CREATE | ALTER | DROP }  [, ...] | ALL [ PRIVILEGES ] }
    ON EXTERNAL SCHEMA schema_name [, ...]
    FROM { IAM_ROLE iam_role } [, ...]
```

Révoquer les autorisations d'unité de partage des données

Autorisations d'unité de partage des données côté producteur

La syntaxe suivante permet d'utiliser REVOKE pour supprimer les autorisations ALTER ou SHARE d'un utilisateur ou d'un rôle. L'utilisateur dont les autorisations ont été révoquées ne peut plus modifier l'unité de partage des données ni en autoriser l'utilisation à un consommateur.

```
REVOKE { ALTER | SHARE } ON DATASHARE datashare_name
 FROM { username [ WITH GRANT OPTION ] | ROLE role_name | GROUP group_name | PUBLIC }
 [, ...]
```

La syntaxe suivante permet d'utiliser REVOKE pour supprimer l'accès d'un consommateur à une unité de partage des données.

```
REVOKE USAGE
 ON DATASHARE datashare_name
 FROM NAMESPACE 'namespaceGUID' [, ...] | ACCOUNT 'accountnumber' [ VIA DATA CATALOG ]
 [, ...]
```

L'exemple suivant montre la révocation de l'utilisation d'une unité de partage des données depuis un compte Lake Formation.

```
REVOKE USAGE ON DATASHARE salesshare FROM ACCOUNT '123456789012' VIA DATA CATALOG;
```

Autorisations d'unité de partage des données côté consommateur

Voici la syntaxe REVOKE pour les autorisations d'utilisation d'unité de partage des données sur une base de données spécifique ou un schéma créé à partir d'une unité de partage des données. La révocation de l'autorisation d'utilisation d'une base de données créée à l'aide de la clause WITH PERMISSIONS ne révoque aucune autorisation supplémentaire que vous avez accordée à un utilisateur ou à un rôle, y compris les autorisations de niveau objet accordées pour les objets sous-jacents. Si vous accordez de nouveau l'autorisation d'utilisation à cet utilisateur ou à ce rôle, il conserve toutes les autorisations supplémentaires dont il disposait avant que vous ne révoquiez l'utilisation.

```
REVOKE USAGE ON { DATABASE shared_database_name [, ...] | SCHEMA shared_schema}
 FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
```

Révocation d'autorisations étendues

Les autorisations étendues vous permettent d'accorder des autorisations à un utilisateur ou à un rôle sur tous les objets d'un type au sein d'une base de données ou d'un schéma. Les utilisateurs et les rôles dotés d'autorisations étendues disposent des autorisations spécifiées sur tous les objets actuels et futurs de la base de données ou du schéma.

Vous pouvez consulter l'étendue des autorisations délimitées au niveau de la base de données dans. SVV_DATABASE_PRIVILEGES Vous pouvez consulter l'étendue des autorisations délimitées au niveau du schéma dans. SVV_SCHEMA_PRIVILEGES

Pour plus d'informations sur les autorisations délimitées, consultezAutorisations étendues.

La syntaxe suivante permet de révoquer des autorisations étendues aux utilisateurs et rôles.

```
REVOKE [ GRANT OPTION ]
{ CREATE | USAGE | ALTER | DROP } [,...] | ALL [ PRIVILEGES ] }
FOR SCHEMAS IN
DATABASE db_name
FROM { username | ROLE role_name } [, ...]
```

```
REVOKE [ GRANT OPTION ]
{ { SELECT | INSERT | UPDATE | DELETE | DROP | ALTER | TRUNCATE | REFERENCES }
 [, ...] } | ALL [PRIVILEGES] } }
FOR TABLES IN
{ SCHEMA schema_name [ DATABASE db_name ] | DATABASE db_name }
FROM { username | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ] { EXECUTE | ALL [ PRIVILEGES ] }
FOR FUNCTIONS IN
{ SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
FROM { username | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ] { EXECUTE | ALL [ PRIVILEGES ] }
FOR PROCEDURES IN
{ SCHEMA schema_name [DATABASE db_name ] | DATABASE db_name }
FROM { username | ROLE role_name } [, ...]

REVOKE [ GRANT OPTION ] USAGE
FOR LANGUAGES IN
DATABASE db_name
FROM { username | ROLE role_name } [, ...]

REVOKE [GRANT_OPTION]
{ { CREATE | ALTER | DROP} [,...] | ALL [ PRIVILEGES ] }
FOR COPY JOBS
IN DATABASE db_name
FROM { username [ WITH GRANT OPTION ] | ROLE role_name } [, ...]
```

Notez que les autorisations délimitées ne font pas de distinction entre les autorisations relatives aux fonctions et aux procédures. Par exemple, l'instruction suivante révoque EXECUTE les autorisations pour les fonctions et les procédures bob du schémaSales_schema.

```
REVOKE EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema FROM bob;
```

Révoquer les autorisations de machine learning

Voici la syntaxe pour les autorisations des modèles de machine learning sur Amazon Redshift.

```
REVOKE [ GRANT OPTION FOR ]
    CREATE MODEL FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
    [ RESTRICT ]

REVOKE [ GRANT OPTION FOR ]
```

```
    { EXECUTE | ALL [ PRIVILEGES ] }
    ON MODEL model_name [, ...]

    FROM { username | ROLE role_name | GROUP group_name | PUBLIC } [, ...]
    [ RESTRICT ]
```

Révoquer les autorisations d'un rôle

Voici la syntaxe permettant de révoquer les autorisations de rôle sur Amazon Redshift.

```
REVOKE [ ADMIN OPTION FOR ] { ROLE role_name } [, ...] FROM { user_name } [, ...]
```

```
REVOKE { ROLE role_name } [, ...] FROM { ROLE role_name } [, ...]
```

Voici la syntaxe permettant de révoquer les autorisations système des rôles sur Amazon Redshift.

```
REVOKE
  {
    { CREATE USER | DROP USER | ALTER USER |
    CREATE SCHEMA | DROP SCHEMA |
    ALTER DEFAULT PRIVILEGES |
    ACCESS CATALOG |
    CREATE TABLE | DROP TABLE | ALTER TABLE |
    CREATE OR REPLACE FUNCTION | CREATE OR REPLACE EXTERNAL FUNCTION |
    DROP FUNCTION |
    CREATE OR REPLACE PROCEDURE | DROP PROCEDURE |
    CREATE OR REPLACE VIEW | DROP VIEW |
    CREATE MODEL | DROP MODEL |
    CREATE DATASHARE | ALTER DATASHARE | DROP DATASHARE |
    CREATE LIBRARY | DROP LIBRARY |
    CREATE ROLE | DROP ROLE
    TRUNCATE TABLE
    VACUUM | ANALYZE | CANCEL }[, ...]
  }
  | { ALL [ PRIVILEGES ] }
FROM { ROLE role_name } [, ...]
```

Révoquer les autorisations d'explication pour les filtres de politique de sécurité au niveau des lignes

Voici la syntaxe permettant de révoquer des autorisations pour expliquer les filtres de politique de sécurité au niveau des lignes d'une requête dans le plan EXPLAIN. Vous pouvez révoquer ce privilège à l'aide de l'instruction REVOKE.

```
REVOKE EXPLAIN RLS FROM ROLE rolename
```

Voici la syntaxe permettant d'accorder des autorisations permettant de contourner les politiques de sécurité au niveau des lignes pour une requête.

```
REVOKE IGNORE RLS FROM ROLE rolename
```

Voici la syntaxe permettant de révoquer les autorisations de la politique de sécurité au niveau des lignes spécifiée.

```
REVOKE SELECT ON [ TABLE ] table_name [, ...]
            FROM RLS POLICY policy_name [, ...]
```

Paramètres

GRANT OPTION FOR

Révoque uniquement l'option d'accorder une autorisation spécifiée à d'autres utilisateurs et ne révoque pas l'autorisation elle-même. Vous ne pouvez pas retirer GRANT OPTION d'un groupe ou de PUBLIC.

SELECT

Révoque l'autorisation de sélectionner les données d'une table ou d'une vue à l'aide d'une instruction SELECT.

INSERT

Révoque l'autorisation de charger les données dans une table à l'aide d'une instruction INSERT ou COPY.

UPDATE

Révoque l'autorisation de mettre à jour une colonne de table à l'aide d'une instruction UPDATE.

DELETE

Révoque l'autorisation de supprimer une ligne d'une table.

REFERENCES

Révoque l'autorisation de créer une contrainte de clé étrangère. Vous devez révoquer cette autorisation sur la table référencée et sur la table qui fait référence.

TRUNCATE

Révoque l'autorisation de tronquer une table. Sans cette autorisation, seul le propriétaire d'une table ou un super-utilisateur peut tronquer une table. Pour plus d'informations sur la commande TRUNCATE, consultez the section called "TRUNCATE".

ALL [PRIVILEGES]

Révoque simultanément toutes les autorisations disponibles de l'utilisateur ou groupe spécifié. Le mot-clé PRIVILEGES est facultatif.

> **Note**
>
> Amazon Redshift ne prend pas en charge les autorisations RULE et TRIGGER. Pour plus d'informations, consultez la section concernant Fonctions PostgreSQL non prises en charge.

ALTER

En fonction de l'objet de la base de données, révoque les autorisations suivantes de l'utilisateur ou du groupe d'utilisateurs :

- Pour les tables, ALTER révoque la modification d'une table ou d'une vue. Pour de plus amples informations, veuillez consulter ALTER TABLE.
- Pour les bases de données, ALTER révoque la modification d'une base de données. Pour de plus amples informations, veuillez consulter ALTER DATABASE.
- Pour les schémas, ALTER révoque la modification d'un schéma. Pour de plus amples informations, veuillez consulter ALTER SCHEMA.
- Pour les tables externes, ALTER révoque la modification d'une table dans un AWS Glue Data Catalog qui est activé pour Lake Formation. Cette autorisation s'applique uniquement lors de l'utilisation de Lake Formation.

DROP

En fonction de l'objet de base de données, révoque les autorisations suivantes accordées à l'utilisateur ou au rôle :

- Pour les tables, DROP révoque l'autorisation de supprimer une table ou une vue. Pour de plus amples informations, veuillez consulter DROP TABLE.

- Pour les bases de données, DROP révoque l'autorisation de supprimer une base de données. Pour de plus amples informations, veuillez consulter DROP DATABASE.
- Pour les schémas, DROP révoque l'autorisation de supprimer un schéma. Pour de plus amples informations, veuillez consulter DROP SCHEMA.

ASSUMEROLE

Révoque l'autorisation d'exécuter les commandes COPY, UNLOAD, EXTERNAL FUNCTION ou CREATE MODEL à partir d'utilisateurs, de rôles ou de groupes possédant un rôle spécifié.

ON [TABLE] nom_table

Révoque les autorisations spécifiées sur une table ou une vue. Le mot-clé TABLE est facultatif.

ON ALL TABLES IN SCHEMA nom_schéma

Révoque les autorisations spécifiées sur toutes les tables du schéma référencé.

(nom_colonne [,...]) ON TABLE nom_table

Révoque les autorisations spécifiées des utilisateurs, groupes ou PUBLIC sur les colonnes spécifiées de la table ou de la vue Amazon Redshift.

(column_list) ON EXTERNAL TABLE schema_name.table_name

Révoque les autorisations spécifiées d'un rôle IAM sur les colonnes spécifiées de la table Lake Formation dans le schéma référencé.

ON EXTERNAL TABLE schema_name.table_name

Révoque les autorisations spécifiées d'un rôle IAM sur les tables Lake Formation spécifiées dans le schéma référencé.

ON EXTERNAL SCHEMA schema_name

Révoque les autorisations spécifiées d'un rôle IAM sur le schéma référencé.

FROM IAM_ROLE iam_role

Indique le rôle IAM qui perd les autorisations.

ROLE role_name

Révoque les autorisations du rôle spécifié.

GROUP group_name

Révoque les autorisations du groupe d'utilisateurs spécifié.

PUBLIC

Révoque les autorisations spécifiées pour tous les utilisateurs. PUBLIC représente un groupe qui inclut toujours tous les utilisateurs. Les autorisations d'un utilisateur sont la somme des autorisations accordées à PUBLIC, des autorisations accordées aux groupes auxquels l'utilisateur appartient et des autorisations accordées à l'utilisateur à titre individuel.

La révocation de PUBLIC d'une table externe Lake Formation entraîne la révocation de l'autorisation du groupe everyone de Lake Formation.

CREATE

En fonction de l'objet de la base de données, révoque les autorisations suivantes de l'utilisateur ou du groupe :

- Pour les bases de données, l'utilisation de la clause CREATE pour REVOKE empêche les utilisateurs de créer des schémas dans la base de données.
- Pour les schémas, l'utilisation de la clause CREATE pour REVOKE empêche les utilisateurs de créer des objets dans un schéma. Pour renommer un objet, l'utilisateur doit disposer de l'autorisation CREATE et posséder l'objet à renommer.

> **Note**
>
> Par défaut, tous les utilisateurs disposent des autorisations CREATE et USAGE sur le schéma PUBLIC.

TEMPORARY | TEMP

Révoque l'autorisation de créer des tables temporaires dans la base de données spécifiée.

> **Note**
>
> Par défaut, les utilisateurs ont l'autorisation de créer des tables temporaires grâce à leur appartenance automatique au groupe PUBLIC. Pour retirer à tout utilisateur l'autorisation de créer des tables temporaires, révoquez l'autorisation TEMP du groupe PUBLIC, puis accordez explicitement l'autorisation de créer des tables temporaires à des utilisateurs ou groupes d'utilisateurs spécifiques.

ON DATABASE nom_db

Révoque les autorisations sur la base de données spécifiée.

USAGE

Révoque les autorisations USAGE sur les objets d'un schéma spécifique, ce qui rend ces objets inaccessibles aux utilisateurs. Les actions spécifiques sur ces objets doivent être révoquées séparément (comme l'autorisation EXECUTE sur les autorisations).

> **Note**
>
> Par défaut, tous les utilisateurs disposent des autorisations CREATE et USAGE sur le schéma PUBLIC.

ON SCHEMA nom_schéma

Révoque les autorisations sur le schéma spécifié. Vous pouvez utiliser les autorisations de schéma pour contrôler la création de tables ; l'autorisation CREATE pour une base de données ne contrôle que la création de schémas.

RESTRICT

Révoque uniquement les autorisations que l'utilisateur a directement accordées. Il s'agit du comportement par défaut.

EXECUTE ON PROCEDURE procedure_name

Révoque l'autorisation EXECUTE sur une procédure stockée spécifique. Comme les noms de procédures stockées peuvent être surchargés, vous devez inclure la liste des arguments de la procédure. Pour plus d'informations, consultez Dénomination des procédures stockées.

EXECUTE ON ALL PROCEDURES IN SCHEMA procedure_name

Révoque les autorisations spécifiées pour toutes les procédures du schéma référencé.

USAGE ON LANGUAGE nom_langage

Révoque l'autorisation d'utilisation d'une langue. Pour les fonctions définies par l'utilisateur en Python (UDFs), utilisez`plpythonu`. Pour SQL UDFs, utilisez`sql`. Pour les procédures stockées, utilisez `plpgsql`.

Pour créer une fonction définie par l'utilisateur, vous devez avoir l'autorisation pour USAGE ON LANGUAGE pour SQL ou `plpythonu` (Python). Par défaut, USAGE ON LANGUAGE SQL

est accordé à PUBLIC. Toutefois, vous devez accorder explicitement USAGE ON LANGUAGE PLPYTHONU à des utilisateurs ou des groupes spécifiques.

Pour révoquer le privilège USAGE pour SQL, révoquez d'abord USAGE de PUBLIC. Accordez ensuite l'utilisation de SQL uniquement aux utilisateurs ou groupes spécifiques autorisés à créer du SQL UDFs. L'exemple suivant révoque le privilège USAGE pour SQL de PUBLIC, puis accorde USAGE au groupe d'utilisateurs `udf_devs`.

```
revoke usage on language sql from PUBLIC;
grant usage on language sql to group udf_devs;
```

Pour plus d'informations, consultez Sécurité et autorisations UDF.

Pour révoquer le privilège USAGE pour les procédures stockées, révoquez d'abord USAGE de PUBLIC. Ensuite, accordez le privilège USAGE sur `plpgsql` uniquement aux utilisateurs ou groupes spécifiques autorisés à créer des procédures stockées. Pour de plus amples informations, veuillez consulter Sécurité et privilèges des procédures stockées .

ON COPY JOB job_name

Révoque les autorisations spécifiées pour une tâche de copie.

FOR { ALL | COPY | UNLOAD | EXTERNAL FUNCTION | CREATE MODEL } [, ...]

Spécifie la commande SQL pour laquelle l'autorisation est révoquée. Vous pouvez spécifier ALL pour révoquer l'autorisation sur les instructions COPY, UNLOAD, EXTERNAL FUNCTION et CREATE MODEL. Cette clause s'applique uniquement à la révocation de l'autorisation ASSUMEROLE.

ALTER

Révoque l'autorisation ALTER pour les utilisateurs ou les groupes d'utilisateurs, ce qui permet à ceux qui ne possèdent pas une unité de partage des données de modifier cette dernière. Cette autorisation est nécessaire pour ajouter ou supprimer des objets d'une unité de partage des données, ou pour définir la propriété PUBLICACCESSIBLE. Pour de plus amples informations, veuillez consulter ALTER DATASHARE.

SHARE

Révoque les autorisations pour les utilisateurs et les groupes d'utilisateurs d'ajouter des consommateurs à une unité de partage des données. La révocation de cette autorisation est nécessaire pour empêcher le consommateur concerné d'accéder à l'unité de partage des données à partir de ses clusters.

ON DATASHARE datashare_name

Accorde les autorisations spécifiées sur l'unité de partage des données référencée.

FROM username

Indique que l'utilisateur a perdu les autorisations.

FROM GROUP group_name

Indique le groupe d'utilisateurs qui perd les autorisations.

WITH GRANT OPTION

Indique que l'utilisateur qui perd les autorisations peut à son tour révoquer les mêmes autorisations pour d'autres personnes. Vous ne pouvez pas révoquer le privilège WITH GRANT OPTION d'un groupe ou de PUBLIC.

USAGE

Lorsque USAGE est révoqué pour un compte consommateurs ou un espace de noms dans le même compte, le compte consommateurs ou l'espace de noms spécifié dans un compte ne peut pas accéder à l'unité de partage des données et aux objets de l'unité de partage des données en lecture seule.

La révocation de l'autorisation d'UTILISATION révoque l'accès des consommateurs à une unité de partage des données.

FROM NAMESPACE 'clusternamespace GUID'

Indique l'espace de noms dans le même compte que celui dont les consommateurs ont perdu les autorisations sur l'unité de partage des données. Les espaces de noms utilisent un identifiant global unique (GUID) alphanumérique de 128 bits.

FROM ACCOUNT 'accountnumber' [VIA DATA CATALOG]

Indique le numéro de compte d'un autre compte dont les consommateurs perdent les autorisations sur le partage des données. La spécification « VIA DATA CATALOG » indique que vous révoquez un compte Lake Formation à utiliser l'unité de partage des données. Si vous omettez le numéro de compte, vous révoquez le compte qui possède le cluster.

ON DATABASE shared_database_name**>** [, ...]

Révoque les autorisations d'utilisation spécifiées sur la base de données spécifiée qui a été créée dans l'unité de partage des données spécifiée.

ON SCHEMA shared_schema

Révoque les autorisations spécifiées sur le schéma spécifié qui a été créé dans l'unité de partage des données spécifiée.

POUR {SCHÉMAS | TABLES | FONCTIONS | PROCÉDURES | LANGUES | TÂCHES DE COPIE} DANS

Spécifie les objets de base de données dont révoquer l'autorisation. Les paramètres situés après IN définissent l'étendue de l'autorisation révoquée.

CREATE MODEL

Révoque l'autorisation CREATE MODEL pour créer des modèles de machine learning dans la base de données spécifiée.

ON MODEL model_name

Révoque l'autorisation EXECUTE pour un modèle spécifique.

ACCESS CATALOG

Révoque l'autorisation d'afficher les métadonnées pertinentes des objets auxquels le rôle a accès.

[ADMIN OPTION FOR] { role } [, ...]

Rôle que vous révoquez pour un utilisateur spécifié disposant de la clause WITH ADMIN OPTION.

FROM { role } [, ...]

Rôle pour lequel vous révoquez le rôle spécifié.

Notes d'utilisation

Pour en savoir plus sur les notes d'utilisation de REVOKE, consultez the section called "Notes d'utilisation".

Exemples

Pour des exemples d'utilisation de REVOKE, consultez the section called "Exemples".

Notes d'utilisation

Pour retirer les privilèges à un objet, vous devez répondre à l'un des critères suivants :

- Être le propriétaire de l'objet.
- Être un super-utilisateur.
- Avoir un privilège accordé pour cet objet et ce privilège.

  Par exemple, la commande suivante autorise l'utilisateur HR à exécuter des commandes SELECT sur la table des employés et à accorder et révoquer le même privilège aux autres utilisateurs.

  ```
  grant select on table employees to HR with grant option;
  ```

  HR ne peut pas révoquer de privilèges pour une opération autre que SELECT, ou sur toute autre table que celle des employés.

Les super-utilisateurs peuvent accéder à tous les objets, quelle que soit les commandes GRANT et REVOKE qui définissent les privilèges d'objet.

PUBLIC représente un groupe qui inclut toujours tous les utilisateurs. Par défaut, tous les membres de PUBLIC disposent des privilèges CREATE et USAGE sur le schéma PUBLIC. Pour limiter toutes les autorisations d'un utilisateur sur le schéma PUBLIC, vous devez d'abord révoquer toutes les autorisations de PUBLIC sur le schéma PUBLIC, puis accorder des privilèges à des utilisateurs ou des groupes spécifiques. L'exemple suivant contrôle les privilèges de création de table du schéma PUBLIC.

```
revoke create on schema public from public;
```

Pour supprimer des privilèges d'une table Lake Formation, le rôle IAM associé au schéma externe de la table doit avoir l'autorisation d'accorder des privilèges à la table externe. L'exemple suivant crée un schéma externe avec un rôle IAM `myGrantor` associé. Le rôle IAM `myGrantor` a l'autorisation de supprimer des autorisations. La commande REVOKE utilise l'autorisation du rôle IAM `myGrantor` associé au schéma externe pour supprimer des autorisations du rôle IAM `myGrantee`.

```
create external schema mySchema
from data catalog
database 'spectrum_db'
iam_role 'arn:aws:iam::123456789012:role/myGrantor'
create external database if not exists;
```

```
revoke select
```

```
on external table mySchema.mytable
from iam_role 'arn:aws:iam::123456789012:role/myGrantee';
```

> **Note**
>
> Si le rôle IAM dispose également d'une ALL AWS Glue Data Catalog autorisation activée pour Lake Formation, l'ALLautorisation n'est pas révoquée. Seule l'autorisation SELECT est supprimée. Vous pouvez afficher les autorisations Lake Formation dans la console Lake Formation.

Notes d'utilisation pour la révocation de l'autorisation ASSUMEROLE

Les notes d'utilisation suivantes s'appliquent à la révocation du privilège ASSUMEROLE dans Amazon Redshift.

Seul un super-utilisateur de base de données peut révoquer le privilège ASSUMEROLE pour les utilisateurs et les groupes. Un super-utilisateur conserve toujours le privilège ASSUMEROLE.

Pour activer l'utilisation du privilège ASSUMEROLE pour les utilisateurs et les groupes, un super-utilisateur exécute l'instruction suivante une fois sur le cluster. Avant d'accorder le privilège ASSUMEROLE aux utilisateurs et aux groupes, un super-utilisateur doit exécuter l'instruction suivante une fois sur le cluster.

```
revoke assumerole on all from public for all;
```

Notes d'utilisation pour la révocation des autorisations de machine learning

Vous ne pouvez pas accorder ou révoquer directement les autorisations liées à une fonction ML. Une fonction ML appartient à un modèle ML et les autorisations sont contrôlées par le biais du modèle. En revanche, vous pouvez révoquer les autorisations liées au modèle ML. L'exemple suivant montre comment révoquer l'autorisation d'exécution de tous les utilisateurs associés au modèle `customer_churn`.

```
REVOKE EXECUTE ON MODEL customer_churn FROM PUBLIC;
```

Vous pouvez également révoquer toutes les autorisations d'un utilisateur pour le modèle ML `customer_churn`.

```
REVOKE ALL on MODEL customer_churn FROM ml_user;
```

L'octroi ou la révocation de l'autorisation EXECUTE liée à une fonction de ML échouera s'il existe une fonction de ML dans le schéma, même si cette fonction de ML dispose déjà de l'autorisation EXECUTE par le biais de GRANT EXECUTE ON MODEL. Nous vous recommandons d'utiliser un schéma distinct lorsque vous utilisez la commande CREATE MODEL afin de conserver les fonctions ML dans un schéma distinct. L'exemple suivant vous montre comment procéder.

```
CREATE MODEL ml_schema.customer_churn
FROM customer_data
TARGET churn
FUNCTION ml_schema.customer_churn_prediction
IAM_ROLE default
SETTINGS (
 S3_BUCKET 'amzn-s3-demo-bucket'
);
```

Exemples

L'exemple suivant retire les privilèges INSERT sur la table SALES au groupe d'utilisateurs GUESTS. Cette commande empêche les membres de GUESTS de pouvoir charger des données dans la table SALES à l'aide de la commande INSERT.

```
revoke insert on table sales from group guests;
```

L'exemple suivant retire le privilège SELECT sur toutes les tables du schéma QA_TICKIT à l'utilisateur fred.

```
revoke select on all tables in schema qa_tickit from fred;
```

L'exemple ci-dessous retire le privilège de sélectionner à partir d'une vue de l'utilisateur bobr.

```
revoke select on table eventview from bobr;
```

L'exemple suivant retire le privilège de créer des tables temporaires dans la base de données TICKIT à tous les utilisateurs.

```
revoke temporary on database tickit from public;
```

L'exemple suivant retire à l'utilisateur cust_name le privilège SELECT sur les colonnes cust_phone et cust_profile de la table user1.

```
revoke select(cust_name, cust_phone) on cust_profile from user1;
```

L'exemple suivant retire au groupe sales_group le privilège SELECT sur les colonnes cust_name et cust_phone et le privilège UPDATE sur la colonne cust_contact_preference de la table cust_profile.

```
revoke select(cust_name, cust_phone), update(cust_contact_preference) on cust_profile
 from group sales_group;
```

L'exemple suivant montre l'utilisation du mot-clé ALL pour retirer au groupe sales_admin les privilèges SELECT et UPDATE sur trois colonnes de la table cust_profile.

```
revoke ALL(cust_name, cust_phone,cust_contact_preference) on cust_profile from group
 sales_admin;
```

L'exemple suivant retire à l'utilisateur cust_name le privilège SELECT sur la colonne cust_profile_vw de la vue user2.

```
revoke select(cust_name) on cust_profile_vw from user2;
```

Exemples de révocation de l'autorisation USAGE des bases de données créées à partir d'unités de partage des données

L'exemple suivant révoque l'accès à l'unité de partage des données salesshare depuis l'espace de noms 13b8833d-17c6-4f16-8fe4-1a018f5ed00d.

```
REVOKE USAGE ON DATASHARE salesshare FROM NAMESPACE
 '13b8833d-17c6-4f16-8fe4-1a018f5ed00d';
```

L'exemple suivant révoque l'autorisation USAGE de Bob sur sales_db.

```
REVOKE USAGE ON DATABASE sales_db FROM Bob;
```

L'exemple suivant révoque l'autorisation USAGE du rôle Analyst_role sur sales_schema.

```
REVOKE USAGE ON SCHEMA sales_schema FROM ROLE Analyst_role;
```

Exemples de révocation d'autorisations étendues

L'exemple suivant révoque l'utilisation pour tous les schémas actuels et futurs de la base de données `Sales_db` pour le rôle `Sales`.

```
REVOKE USAGE FOR SCHEMAS IN DATABASE Sales_db FROM ROLE Sales;
```

L'exemple suivant révoque la possibilité d'accorder à l'utilisateur `alice` l'autorisation SELECT pour toutes les tables actuelles et futures de la base de données `Sales_db`. `alice` conserve l'accès à toutes les tables de `Sales_db`.

```
REVOKE GRANT OPTION SELECT FOR TABLES IN DATABASE Sales_db FROM alice;
```

L'exemple suivant révoque l'autorisation EXECUTE accordée à l'utilisateur bob pour les fonctions du schéma `Sales_schema`.

```
REVOKE EXECUTE FOR FUNCTIONS IN SCHEMA Sales_schema FROM bob;
```

L'exemple suivant révoque toutes les autorisations accordées au rôle `Sales` pour toutes les tables du schéma `ShareSchema` de la base de données `ShareDb`. Lorsque vous spécifiez le schéma, vous pouvez aussi spécifier la base de données du schéma en utilisant le format en deux parties `database.schema`.

```
REVOKE ALL FOR TABLES IN SCHEMA ShareDb.ShareSchema FROM ROLE Sales;
```

L'exemple suivant est le même que le précédent. Vous pouvez spécifier la base de données du schéma à l'aide du mot-clé `DATABASE` au lieu d'utiliser un format en deux parties.

```
REVOKE ALL FOR TABLES IN SCHEMA ShareSchema DATABASE ShareDb FROM ROLE Sales;
```

Exemples de révocation du privilège ASSUMEROLE

Voici des exemples de révocation du privilège ASSUMEROLE.

Un super-utilisateur doit activer l'utilisation du privilège ASSUMEROLE pour les utilisateurs et les groupes en exécutant l'instruction suivante une fois sur le cluster :

```
revoke assumerole on all from public for all;
```

L'instruction suivante révoque le privilège ASSUMEROLE de l'utilisateur reg_user1 sur tous les rôles pour toutes les opérations.

```
revoke assumerole on all from reg_user1 for all;
```

Exemples de révocation du privilège ROLE

L'exemple suivant révoque le rôle sample_role1 pour le rôle sample_role2.

```
CREATE ROLE sample_role2;
GRANT ROLE sample_role1 TO ROLE sample_role2;
REVOKE ROLE sample_role1 FROM ROLE sample_role2;
```

L'exemple suivant révoque les privilèges système pour l'utilisateur user1.

```
GRANT ROLE sys:DBA TO user1;
REVOKE ROLE sys:DBA FROM user1;
```

L'exemple suivant révoque les rôles sample_role1 et sample_role2 pour l'utilisateur user1.

```
CREATE ROLE sample_role1;
CREATE ROLE sample_role2;
GRANT ROLE sample_role1, ROLE sample_role2 TO user1;
REVOKE ROLE sample_role1, ROLE sample_role2 FROM user1;
```

L'exemple suivant révoque le rôle sample_role2 avec la clause WITH ADMIN OPTION pour l'utilisateur user1.

```
GRANT ROLE sample_role2 TO user1 WITH ADMIN OPTION;
REVOKE ADMIN OPTION FOR ROLE sample_role2 FROM user1;
REVOKE ROLE sample_role2 FROM user1;
```

L'exemple suivant révoque les rôles sample_role1 et sample_role2 pour le rôle sample_role5.

```
CREATE ROLE sample_role5;
GRANT ROLE sample_role1, ROLE sample_role2 TO ROLE sample_role5;
```

```
REVOKE ROLE sample_role1, ROLE sample_role2 FROM ROLE sample_role5;
```

L'exemple suivant révoque les privilèges système CREATE SCHEMA et DROP SCHEMA pour le rôle sample_role1.

```
GRANT CREATE SCHEMA, DROP SCHEMA TO ROLE sample_role1;
REVOKE CREATE SCHEMA, DROP SCHEMA FROM ROLE sample_role1;
```

ROLLBACK

Arrête la transaction en cours et ignore toutes les mises à jour effectuées par cette transaction.

Cette commande effectue la même fonction que la commande ABORT.

Syntaxe

```
ROLLBACK [ WORK | TRANSACTION ]
```

Paramètres

WORK

Mot-clé facultatif. Ce mot-clé n'est pas pris en charge dans une procédure stockée.

TRANSACTION

Mot-clé facultatif. WORK et TRANSACTION sont synonymes. Aucun des deux n'est pris en charge dans une procédure stockée.

Pour obtenir des informations sur l'utilisation de ROLLBACK dans une procédure stockée, consultez Gestion des transactions.

exemple

L'exemple suivant crée une table, puis démarre une transaction où les données sont insérées dans la table. La commande ROLLBACK annule ensuite l'insertion des données et laisse la table vide.

La commande suivante crée un exemple de table appelée MOVIE_GROSS :

```
create table movie_gross( name varchar(30), gross bigint );
```

La prochaine série de commandes démarre une transaction qui insère deux lignes de données dans la table :

```
begin;

insert into movie_gross values ( 'Raiders of the Lost Ark', 23400000);

insert into movie_gross values ( 'Star Wars', 10000000 );
```

Puis, la commande suivante sélectionne les données de la table pour montrer qu'elles ont été correctement insérées :

```
select * from movie_gross;
```

La sortie de la commande montre que les deux lignes ont été insérées avec succès :

```
name           |  gross
-------------------------+----------
Raiders of the Lost Ark | 23400000
Star Wars               | 10000000
(2 rows)
```

Cette commande restaure les modifications des données à l'emplacement où la transaction a commencé :

```
rollback;
```

La sélection de données de la table affiche maintenant une table vide :

```
select * from movie_gross;

name | gross
------+-------
(0 rows)
```

SELECT

renvoie les lignes des tables, vues et fonctions définies par l'utilisateur.

> **Note**
>
> La taille maximale d'une instruction SQL est de 16 Mo.

Syntaxe

```
[ WITH with_subquery [, ...] ]
SELECT
[ TOP number | [ ALL | DISTINCT ]
* | expression [ AS output_name ] [, ...] ]
[ EXCLUDE column_list ]
[ FROM table_reference [, ...] ]
[ WHERE condition ]
[ [ START WITH expression ] CONNECT BY expression ]
[ GROUP BY ALL | expression [, ...] ]
[ HAVING condition ]
[ QUALIFY condition ]
[ { UNION | ALL | INTERSECT | EXCEPT | MINUS } query ]
[ ORDER BY expression [ ASC | DESC ] ]
[ LIMIT { number | ALL } ]
[ OFFSET start ]
```

Rubriques

- Clause WITH
- Liste SELECT
- EXCLURE column_list
- Clause FROM
- Clause WHERE
- Clause GROUP BY
- Clause HAVING
- Clause QUALIFY
- UNION, INTERSECT et EXCEPT
- Clause ORDER BY
- Clause CONNECT BY
- Exemples de sous-requête

- Sous-requêtes corrélées

Clause WITH

Une clause WITH est une clause facultative qui précède la liste SELECT d'une requête. La clause WITH définit une ou plusieurs expressions common_table_expressions. Chaque expression de table commune (CTE) définit une table temporaire, qui est similaire à la définition d'une vue. Vous pouvez référencer ces tables temporaires dans la clause FROM. Elles ne sont utilisées que pendant l'exécution de la requête à laquelle elles appartiennent. Chaque CTE de la clause WITH spécifie un nom de table, une liste facultative de noms de colonne et une expression de requête correspondant à une table (instruction SELECT). Lorsque vous référencez le nom de la table temporaire dans la clause FROM de la même expression de requête qui la définit, la CTE est récursive.

Les sous-requêtes de clause WITH sont un moyen efficace de définir les tables qui peuvent être utilisées tout au long de l'exécution d'une même requête. Dans tous les cas, les mêmes résultats peuvent être obtenus à l'aide de sous-requêtes dans le corps principal de l'instruction SELECT, mais les sous-requêtes de clause WITH peuvent être plus simples à lire et à écrire. Chaque fois que possible, les sous-requêtes de clause WITH qui sont référencées plusieurs fois sont optimisées en tant que sous-expressions courantes ; autrement dit, il peut être possible d'évaluer une sous-requête WITH une fois et de réutiliser ses résultats. (Notez que les sous-expressions courantes ne sont pas limitées à celles définies dans la clause WITH).

Syntaxe

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
```

où common_table_expression peut être récursive ou non-récursive. Voici la forme non-récursive :

```
CTE_table_name [ ( column_name [, ...] ) ] AS ( query )
```

Voici la forme récursive de common_table_expression :

```
CTE_table_name (column_name [, ...] ) AS ( recursive_query )
```

Paramètres

RECURSIVE

Mot-clé qui identifie la requête comme étant une CTE récursive. Ce mot-clé est requis si l'expression common_table_expression définie dans la clause WITH est récursive. Vous ne pouvez spécifier le mot clé RECURSIVE qu'une seule fois, immédiatement après le mot clé WITH, même si la clause WITH contient plusieurs CTEs récursifs. En général, une CTE récursive est une sous-requête UNION ALL avec deux parties.

common_table_expression

Définit une table temporaire que vous pouvez référencer dans Clause FROM et qui n'est utilisée que pendant l'exécution de la requête à laquelle elle appartient.

CTE_table_name

Nom unique d'une table temporaire qui définit les résultats d'une sous-requête de clause WITH. Vous ne pouvez pas utiliser de noms en double au sein d'une clause WITH. Chaque sous-requête doit avoir un nom de table qui peut être référencé dans la Clause FROM.

column_name

Liste des noms de colonne de sortie pour la sous-requête de clause WITH, séparés par des virgules. Le nombre de noms de colonne spécifié doit être égal ou inférieur au nombre de colonnes défini par la sous-requête. Pour une CTE non récursive, column_name est facultatif. Pour une CTE récursive, column_name est obligatoire.

query

Toute requête SELECT qu'Amazon Redshift prend en charge. Consultez SELECT.

recursive_query

Requête UNION ALL qui se compose de deux sous-requêtes SELECT :

- La première sous-requête SELECT n'a pas de référence récursive à la même table CTE_table_name. Elle renvoie un ensemble de résultats qui est la base initiale de la récursivité. Cette partie est appelée initial member ou seed member.
- La deuxième sous-requête SELECT fait référence à la même table CTE_table_name dans sa clause FROM. C'est ce qu'on appelle le recursive member. recursive_query contient une condition WHERE pour mettre fin à la recursive_query.

Notes d’utilisation

Vous pouvez utiliser une clause WITH dans les instructions SQL suivantes :

- SELECT
- SELECT INTO
- CREATE TABLE AS
- CREATE VIEW
- DECLARE
- EXPLAIN
- INSERT INTO...SELECT
- PREPARE
- UPDATE (dans une sous-requête de clause WHERE. Vous ne pouvez pas définir une CTE récursive dans la sous-requête. La CTE récursive doit précéder la clause UPDATE.)
- DELETE

Si la clause FROM d’une requête qui contient une clause WITH ne fait pas référence à l’une des tables définies par la clause WITH, la clause WITH est ignorée et la requête s’exécute normalement.

Une table définie par une sous-requête de clause WITH peut être référencée uniquement dans la portée de la requête SELECT que commence la clause WITH. Par exemple, vous pouvez faire référence à une telle table dans la clause FROM d’une sous-requête de la liste SELECT, la clause WHERE ou la clause HAVING. Vous ne pouvez pas utiliser une clause WITH dans une sous-requête et faire référence à sa table dans la clause FROM de la requête principale ou d’une autre sous-requête. Ce modèle de requête entraîne un message d’erreur sous la forme `relation table_name doesn't exist` pour la table de la clause WITH.

Vous ne pouvez pas spécifier une autre clause WITH à l’intérieur d’une sous-requête de clause WITH.

Vous ne pouvez pas effectuer de références futures aux tables définies par des sous-requêtes de clause WITH. Par exemple, la requête suivante renvoie une erreur en raison de la référence future à la table W2 dans la définition de table W1 :

```
with w1 as (select * from w2), w2 as (select * from w1)
select * from sales;
```

```
ERROR:  relation "w2" does not exist
```

Une sous-requête de clause WITH peut ne pas comporter d'instruction SELECT INTO ; cependant, vous pouvez utiliser une clause WITH dans une instruction SELECT INTO.

Expressions récursives de table commune

Une expression de table commune (CTE) récursive est une CTE qui se réfère à elle-même. Une CTE récursive est utile pour interroger des données hiérarchiques, telles que des organigrammes qui montrent les rapports hiérarchiques entre les employés et les responsables. Consultez Exemple : CTE récursive.

Une autre utilisation courante est une nomenclature à plusieurs niveaux, lorsqu'un produit est constitué de nombreux composants et que chaque composant lui-même est également constitué d'autres composants ou sous-ensembles.

Veillez à limiter la profondeur de récursivité en incluant une clause WHERE dans la deuxième sous-requête SELECT de la requête récursive. Pour obtenir un exemple, consultez Exemple : CTE récursive. Sinon, une erreur similaire à ce qui suit peut se produire :

- `Recursive CTE out of working buffers.`
- `Exceeded recursive CTE max rows limit, please add correct CTE termination predicates or change the max_recursion_rows parameter.`

> **Note**
>
> `max_recursion_rows` est un paramètre définissant le nombre maximal de lignes qu'une CTE récursive peut renvoyer afin d'éviter les boucles de récursion infinies. Nous vous recommandons de ne pas modifier cette valeur pour qu'elle soit supérieure à la valeur par défaut. Cela permet d'éviter que les problèmes de récursion infinie dans vos requêtes n'occupent trop d'espace dans votre cluster.

Vous pouvez spécifier un ordre de tri et une limite sur le résultat de la CTE récursive. Vous pouvez inclure des options group by et distinct sur le résultat final de la CTE récursive.

Vous ne pouvez pas spécifier une autre clause WITH RECURSIVE dans une sous-requête. Le membre recursive_query ne peut pas inclure de clause order by ou limit.

Exemples

L'exemple suivant illustre le cas le plus simple possible d'une requête contenant une clause WITH. La requête WITH nommée VENUECOPY sélectionne toutes les lignes de la table VENUE. La requête principale, à son tour, sélectionne toutes les lignes de VENUECOPY. La table VENUECOPY existe uniquement pendant la durée de cette requête.

```
with venuecopy as (select * from venue)
select * from venuecopy order by 1 limit 10;
```

```
 venueid |         venuename          |    venuecity    | venuestate | venueseats
---------+----------------------------+-----------------+------------+------------
1 | Toyota Park                | Bridgeview      | IL         |          0
2 | Columbus Crew Stadium      | Columbus        | OH         |          0
3 | RFK Stadium                | Washington      | DC         |          0
4 | CommunityAmerica Ballpark  | Kansas City     | KS         |          0
5 | Gillette Stadium           | Foxborough      | MA         |      68756
6 | New York Giants Stadium    | East Rutherford | NJ         |      80242
7 | BMO Field                  | Toronto         | ON         |          0
8 | The Home Depot Center      | Carson          | CA         |          0
9 | Dick's Sporting Goods Park | Commerce City   | CO         |          0
v     10 | Pizza Hut Park             | Frisco          | TX         |          0
(10 rows)
```

L'exemple suivant montre une clause WITH qui produit deux tables, nommées VENUE_SALES et TOP_VENUES. La deuxième table de requête WITH effectue la sélection à partir de la première. A son tour, la clause WHERE du bloc de requête principal contient une sous-requête qui restreint la table TOP_VENUES.

```
with venue_sales as
(select venuename, venuecity, sum(pricepaid) as venuename_sales
from sales, venue, event
where venue.venueid=event.venueid and event.eventid=sales.eventid
group by venuename, venuecity),

top_venues as
(select venuename
from venue_sales
where venuename_sales > 800000)

select venuename, venuecity, venuestate,
```

```
sum(qtysold) as venue_qty,
sum(pricepaid) as venue_sales
from sales, venue, event
where venue.venueid=event.venueid and event.eventid=sales.eventid
and venuename in(select venuename from top_venues)
group by venuename, venuecity, venuestate
order by venuename;
```

```
        venuename       |   venuecity   | venuestate | venue_qty | venue_sales
------------------------+---------------+------------+-----------+-------------
August Wilson Theatre   | New York City | NY         |      3187 |  1032156.00
Biltmore Theatre        | New York City | NY         |      2629 |   828981.00
Charles Playhouse       | Boston        | MA         |      2502 |   857031.00
Ethel Barrymore Theatre | New York City | NY         |      2828 |   891172.00
Eugene O'Neill Theatre  | New York City | NY         |      2488 |   828950.00
Greek Theatre           | Los Angeles   | CA         |      2445 |   838918.00
Helen Hayes Theatre     | New York City | NY         |      2948 |   978765.00
Hilton Theatre          | New York City | NY         |      2999 |   885686.00
Imperial Theatre        | New York City | NY         |      2702 |   877993.00
Lunt-Fontanne Theatre   | New York City | NY         |      3326 |  1115182.00
Majestic Theatre        | New York City | NY         |      2549 |   894275.00
Nederlander Theatre     | New York City | NY         |      2934 |   936312.00
Pasadena Playhouse      | Pasadena      | CA         |      2739 |   820435.00
Winter Garden Theatre   | New York City | NY         |      2838 |   939257.00
(14 rows)
```

Les deux exemples suivants illustrent les règles sur la portée des références de table dans les sous-requêtes de clause WITH. La première requête s'exécute, mais la deuxième échoue avec une erreur prévue. La première requête a une sous-requête de clause WITH à l'intérieur de la liste SELECT de la requête principale. La table définie par la clause WITH (HOLIDAYS) est référencée dans la clause FROM de la sous-requête de la liste SELECT :

```
select caldate, sum(pricepaid) as daysales,
(with holidays as (select * from date where holiday ='t')
select sum(pricepaid)
from sales join holidays on sales.dateid=holidays.dateid
where caldate='2008-12-25') as dec25sales
from sales join date on sales.dateid=date.dateid
where caldate in('2008-12-25','2008-12-31')
group by caldate
order by caldate;
```

```
caldate   | daysales | dec25sales
-----------+----------+------------
2008-12-25 | 70402.00 |   70402.00
2008-12-31 | 12678.00 |   70402.00
(2 rows)
```

La deuxième requête échoue, car elle tente de faire référence à la table HOLIDAYS de la requête principale, ainsi que dans la sous-requête de liste SELECT. Les références de requête principale sont hors de portée.

```
select caldate, sum(pricepaid) as daysales,
(with holidays as (select * from date where holiday ='t')
select sum(pricepaid)
from sales join holidays on sales.dateid=holidays.dateid
where caldate='2008-12-25') as dec25sales
from sales join holidays on sales.dateid=holidays.dateid
where caldate in('2008-12-25','2008-12-31')
group by caldate
order by caldate;

ERROR:  relation "holidays" does not exist
```

Exemple : CTE récursive

Voici un exemple de CTE récursive qui renvoie les employés qui relèvent directement ou indirectement de John. La requête récursive contient une clause WHERE pour limiter la profondeur de récursivité à moins de 4 niveaux.

```
--create and populate the sample table
  create table employee (
  id int,
  name varchar (20),
  manager_id int
  );

  insert into employee(id, name, manager_id)  values
(100, 'Carlos', null),
(101, 'John', 100),
(102, 'Jorge', 101),
(103, 'Kwaku', 101),
(110, 'Liu', 101),
(106, 'Mateo', 102),
```

```
(110, 'Nikki', 103),
(104, 'Paulo', 103),
(105, 'Richard', 103),
(120, 'Saanvi', 104),
(200, 'Shirley', 104),
(201, 'Sofía', 102),
(205, 'Zhang', 104);

--run the recursive query
  with recursive john_org(id, name, manager_id, level) as
( select id, name, manager_id, 1 as level
  from employee
  where name = 'John'
  union all
  select e.id, e.name, e.manager_id, level + 1 as next_level
  from employee e, john_org j
  where e.manager_id = j.id and level < 4
  )
 select distinct id, name, manager_id from john_org order by manager_id;
```

Voici le résultat de la requête.

```
    id        name      manager_id
  ------+-----------+--------------
   101    John           100
   102    Jorge          101
   103    Kwaku          101
   110    Liu            101
   201    Sofía          102
   106    Mateo          102
   110    Nikki          103
   104    Paulo          103
   105    Richard        103
   120    Saanvi         104
   200    Shirley        104
   205    Zhang          104
```

Voici un organigramme du service de John.

Liste SELECT

Rubriques

- Syntaxe
- Paramètres
- Notes d'utilisation
- Exemples

La liste SELECT nomme les colonnes, fonctions et expressions que la requête doit renvoyer. La liste représente le résultat de la requête.

Pour plus d'informations sur les fonctions SQL, consultez Référence sur les fonctions SQL. Pour plus d'informations sur les expressions, consultez Expressions conditionnelles.

Syntaxe

```
SELECT
[ TOP number ]
[ ALL | DISTINCT ] * | expression [ AS column_alias ] [, ...]
```

Paramètres

TOP nombre

TOP accepte un nombre entier positif comme argument, qui définit le nombre de lignes retournées au client. Le comportement avec la clause TOP est identique à celui avec la clause LIMIT. Le nombre de lignes qui est renvoyé est fixe, contrairement à l'ensemble de lignes. Pour renvoyer un ensemble de lignes constant, utilisez TOP ou LIMIT avec une clause ORDER BY.

ALL

Mot-clé redondant qui définit le comportement par défaut, si vous ne spécifiez pas DISTINCT. `SELECT ALL *` signifie la même chose que `SELECT *` (sélectionner toutes les lignes de toutes les colonnes et conserver les doublons).

DISTINCT

Option qui élimine les lignes en double du jeu de résultats, en fonction de la correspondance des valeurs dans une ou plusieurs colonnes.

> **Note**
>
> Si votre application autorise des clés étrangères ou des clés primaires non valides, les requêtes peuvent renvoyer des résultats incorrects. Par exemple, une requête

SELECT DISTINCT peut renvoyer des lignes dupliquées si la colonne de clé primaire ne contient pas toutes les valeurs uniques. Pour plus d'informations, consultez Définition des contraintes de table.

* (astérisque)

renvoie tout le contenu de la table (toutes les colonnes et toutes les lignes).

expression

Expression formée d'une ou de plusieurs colonnes qui existent dans les tables référencées par la requête. Une expression peut contenir des fonctions SQL. Par exemple :

```
avg(datediff(day, listtime, saletime))
```

AS alias_colonne

Nom temporaire de la colonne utilisé dans le jeu de résultats final. Le mot-clé AS est facultatif. Par exemple :

```
avg(datediff(day, listtime, saletime)) as avgwait
```

Si vous ne spécifiez pas un alias pour une expression qui n'est pas un nom de colonne simple, le jeu de résultats applique un nom par défaut à cette colonne.

> **Note**
>
> L'alias est reconnu juste après sa définition dans la liste cible. Vous pouvez utiliser un alias dans d'autres expressions définies après lui dans la même liste cible. L'exemple suivant illustre ce scénario.
>
> ```
> select clicks / impressions as probability, round(100 * probability, 1) as
>  percentage from raw_data;
> ```
>
> La référence latérale à un alias vous évite de devoir répéter l'expression disposant d'un alias lors de la création d'expressions plus complexes dans la même liste cible.

> Lorsqu'Amazon Redshift analyse ce type de référence, il rajoute simplement dans la ligne les alias précédemment définis. S'il existe une colonne avec le même nom défini dans la clause FROM que dans l'expression disposant d'un alias précédente, la colonne dans la clause FROM a la priorité. Par exemple, dans la requête précédente, s'il existe une colonne nommée « probability » dans la table raw_data, le terme « probability » dans la seconde expression de la liste cible fait référence à cette colonne et non au nom d'alias « probability ».

Notes d'utilisation

TOP est une extension SQL ; elle fournit une alternative au comportement LIMIT. Vous ne pouvez pas utiliser TOP et LIMIT dans la même requête.

Exemples

L'exemple suivant renvoie 10 lignes de la table SALES. Bien que la requête utilise la clause TOP, elle renvoie toujours un ensemble imprévisible de lignes, car aucune clause ORDER BY n'est spécifiée.

```
select top 10 *
from sales;
```

La requête suivante est fonctionnellement équivalente, mais utilise une clause LIMIT au lieu d'une clause TOP :

```
select *
from sales
limit 10;
```

L'exemple suivant renvoie les 10 premières lignes de la table SALES en utilisant la clause TOP, classées dans la colonne QTYSOLD par ordre décroissant.

```
select top 10 qtysold, sellerid
from sales
order by qtysold desc, sellerid;

qtysold | sellerid
--------+----------
8 |      518
```

```
8 |      520
8 |      574
8 |      718
8 |      868
8 |     2663
8 |     3396
8 |     3726
8 |     5250
8 |     6216
(10 rows)
```

L'exemple suivant renvoie les deux premières valeurs QTYSOLD et SELLERID de la table SALES, classées par la colonne QTYSOLD :

```
select top 2 qtysold, sellerid
from sales
order by qtysold desc, sellerid;

qtysold | sellerid
--------+----------
8 |      518
8 |      520
(2 rows)
```

L'exemple suivant montre la liste des groupes de catégories distincts de la table CATEGORY :

```
select distinct catgroup from category
order by 1;

catgroup
----------
Concerts
Shows
Sports
(3 rows)

--the same query, run without distinct
select catgroup from category
order by 1;

catgroup
----------
Concerts
```

```
Concerts
Concerts
Shows
Shows
Shows
Sports
Sports
Sports
Sports
Sports
(11 rows)
```

L'exemple suivant renvoie l'ensemble distinct de numéros de semaine pour décembre 2008. Sans la clause DISTINCT, l'instruction renverrait 31 lignes, soit une pour chaque jour du mois.

```
select distinct week, month, year
from date
where month='DEC' and year=2008
order by 1, 2, 3;

week | month | year
-----+-------+------
49 | DEC   | 2008
50 | DEC   | 2008
51 | DEC   | 2008
52 | DEC   | 2008
53 | DEC   | 2008
(5 rows)
```

EXCLURE column_list

La liste de colonnes EXCLUDE nomme les colonnes exclues des résultats de la requête. L'utilisation de l'option EXCLUDE est utile lorsque seul un sous-ensemble de colonnes doit être exclu d'un tableau large, c'est-à-dire un tableau contenant de nombreuses colonnes.

Rubriques

- Syntaxe
- Paramètres
- Exemples

Syntaxe

```
EXCLUDE column_list
```

Paramètres

column_list

Liste séparée par des virgules d'un ou de plusieurs noms de colonnes figurant dans les tables référencées par la requête. La column_list peut éventuellement être placée entre parenthèses. Seuls les noms de colonnes sont pris en charge dans la liste d'exclusion des noms de colonnes, et non les expressions (telles que`upper(col1)`) ou les astérisques (*).

```
column-name, ... | ( column-name, ... )
```

Par exemple :

```
SELECT * EXCLUDE col1, col2 FROM tablea;
```

```
SELECT * EXCLUDE (col1, col2) FROM tablea;
```

Exemples

Les exemples suivants utilisent la table SALES qui contient des colonnes : salesid, listid, sellerid, buyerid, eventid, dateid, qtysold, pricepaid, commission et saletime. Pour plus d'informations sur le tableau SALES, consultezExemple de base de données.

L'exemple suivant renvoie des lignes de la table SALES, mais exclut la colonne SALETIME.

```
SELECT * EXCLUDE saletime FROM sales;

salesid | listid  | sellerid | buyerid | eventid | dateid  | qtysold  | pricepaid  |
 commission
--------+---------+----------+---------+---------+---------+----------+------------
+-----------
150314  | 173969  | 48680    | 816     | 8762    | 1827    | 2        | 688        |
 103.2
8325    | 8942    | 23600    | 1078    | 2557    | 1828    | 5        | 525        |
 78.75
```

```
46807   | 52711   | 34388    | 1047    | 2046    | 1828    | 2        | 482        |
 72.3
...
```

L'exemple suivant renvoie des lignes de la table SALES, mais exclut les colonnes QTYSOLD et SALETIME.

```
SELECT * EXCLUDE (qtysold, saletime) FROM sales;

salesid | listid  | sellerid | buyerid | eventid | dateid  | pricepaid  | commission
--------+---------+----------+---------+---------+---------+------------+-----------
150314  | 173969  | 48680    | 816     | 8762    | 1827    | 688        | 103.2
8325    | 8942    | 23600    | 1078    | 2557    | 1828    | 525        |  78.75
46807   | 52711   | 34388    | 1047    | 2046    | 1828    | 482        |  72.3
...
```

L'exemple suivant crée une vue qui renvoie des lignes de la table SALES, mais exclut la colonne SALETIME.

```
CREATE VIEW sales_view AS SELECT * EXCLUDE saletime FROM sales;
SELECT * FROM sales_view;

salesid | listid  | sellerid | buyerid | eventid | dateid  | qtysold  | pricepaid  |
 commission
--------+---------+----------+---------+---------+---------+----------+------------
+-----------
150314  | 173969  | 48680    | 816     | 8762    | 1827    | 2        | 688        |
 103.2
8325    | 8942    | 23600    | 1078    | 2557    | 1828    | 5        | 525        |
 78.75
46807   | 52711   | 34388    | 1047    | 2046    | 1828    | 2        | 482        |
 72.3
...
```

L'exemple suivant sélectionne uniquement les colonnes qui ne sont pas exclues d'une table temporaire.

```
SELECT * EXCLUDE saletime INTO TEMP temp_sales FROM sales;
SELECT * FROM temp_sales;

salesid | listid  | sellerid | buyerid | eventid | dateid  | qtysold  | pricepaid  |
 commission
```

```
--------+---------+----------+---------+---------+---------+----------+------------
+-----------
150314  | 173969  | 48680    | 816     | 8762    | 1827    | 2        | 688        |
 103.2
8325    | 8942    | 23600    | 1078    | 2557    | 1828    | 5        | 525        |
 78.75
46807   | 52711   | 34388    | 1047    | 2046    | 1828    | 2        | 482        |
 72.3
...
```

Clause FROM

La clause FROM d'une requête répertorie les références de table (tables, vues et sous-requêtes) à partir desquelles les données sont sélectionnées. Si plusieurs références de table sont répertoriées, les tables doivent être jointes, à l'aide de la syntaxe appropriée de la clause FROM ou de la clause WHERE. Si aucun critère de jointure n'est spécifié, le système traite la requête comme jointure croisée (produit cartésien).

Rubriques

- Syntaxe
- Paramètres
- Notes d'utilisation
- Exemples PIVOT et UNPIVOT
- Exemples de clause JOIN

Syntaxe

```
FROM table_reference [, ...]
```

où table_reference est l'une des références suivantes :

```
with_subquery_table_name [ table_alias ]
table_name [ * ] [ table_alias ]
( subquery ) [ table_alias ]
table_reference [ NATURAL ] join_type table_reference
   [ ON join_condition | USING ( join_column [, ...] ) ]
table_reference PIVOT (
   aggregate(expr) [ [ AS ] aggregate_alias ]
   FOR column_name IN ( expression [ AS ] in_alias [, ...] )
```

```
) [ table_alias ]
table_reference UNPIVOT [ INCLUDE NULLS | EXCLUDE NULLS ] (
   value_column_name
   FOR name_column_name IN ( column_reference [ [ AS ]
   in_alias ] [, ...] )
) [ table_alias ]
UNPIVOT expression AS value_alias [ AT attribute_alias ]
```

L'option table_alias peut être utilisée pour donner des noms temporaires aux tables et aux références de tables complexes et, si vous le souhaitez, à leurs colonnes également, comme suit :

```
[ AS ] alias [ ( column_alias [, ...] ) ]
```

Paramètres

with_subquery_table_name

Table définie par une sous-requête dans la Clause WITH.

table_name

Nom d'une table ou d'une vue.

alias

Nom alternatif temporaire d'une table ou d'une vue. Un alias doit être fourni pour une table dérivée d'une sous-requête. Dans les autres références de table, les alias sont facultatifs. Le mot-clé AS est toujours facultatif. Les alias de table offrent un raccourci pratique pour identifier les tables dans d'autres parties d'une requête, telles que la clause WHERE. Par exemple :

```
select * from sales s, listing l
where s.listid=l.listid
```

alias_colonne

Nom alternatif temporaire pour une colonne dans une table ou une vue.

sous-requête

Une expression de requête qui correspond à une table. La table existe uniquement pendant la durée de la requête et reçoit généralement un nom ou un alias. Toutefois, l'alias n'est pas obligatoire. Vous pouvez aussi définir des noms de colonnes pour les tables qui proviennent de sous-requêtes. Il est important de nommer les alias de colonne lorsque vous souhaitez joindre les

résultats des sous-requêtes à d'autres tables et lorsque vous voulez sélectionner ou limiter les colonnes ailleurs dans la requête.

Une sous-requête peut contenir une clause ORDER BY, mais cette clause peut n'avoir aucun effet si une clause LIMIT ou OFFSET n'est pas également spécifiée.

NATURAL

Définit une jointure qui utilise automatiquement toutes les paires de colonnes portant le même nom dans les deux tables comme colonnes de jointure. Aucune condition de jointure explicite n'est nécessaire. Par exemple, si les tables CATEGORY et EVENT ont toutes deux des colonnes nommées CATID, une jointure naturelle des tables est une jointure sur leurs colonnes CATID.

> **Note**
>
> Si une jointure NATURAL est spécifiée, mais qu'il n'y a aucune paire de colonnes portant le même nom dans les tables à joindre, la requête se résout par défaut en une jointure croisée.

join_type

Spécifiez l'un des types de jointure suivants :

- [INNER] JOIN
- LEFT [OUTER] JOIN
- RIGHT [OUTER] JOIN
- FULL [OUTER] JOIN
- CROSS JOIN

Les jointures croisées sont des jointures non qualifiées ; elles renvoient le produit cartésien des deux tables.

Les jointures internes et externes sont des jointures qualifiées. Elles sont qualifiés implicitement (en jointures naturelles), avec la syntaxe ON ou USING de la clause FROM, ou avec une condition de clause WHERE.

Une jointure interne renvoie les lignes correspondantes uniquement, en fonction de la condition de jointure ou d'une liste de colonnes de jointure. Une jointure externe renvoie toutes les lignes que la jointure interne équivalente renvoyerait, plus les lignes non correspondantes de la table de

« gauche », de la table de « droite » ou des deux tables. La table de gauche est la première table de la liste et la table de droite la deuxième table. Les lignes non correspondantes contiennent des valeurs NULL pour combler les écarts dans les colonnes de sortie.

ON condition_jointure

Type de spécification de jointure où les colonnes de jointure sont définies comme condition qui suit le mot-clé ON. Par exemple :

```
sales join listing
on sales.listid=listing.listid and sales.eventid=listing.eventid
```

USING (colonne_jointure [, ...])

Type de spécification de jointure où les colonnes de jointure sont affichées entre parenthèses. Si plusieurs colonnes de jointure sont spécifiées, elles sont séparées par des virgules. Le mot-clé USING doit précéder la liste. Par exemple :

```
sales join listing
using (listid,eventid)
```

PIVOT

Fait pivoter la sortie des lignes vers les colonnes, dans le but de représenter les données tabulaires dans un format facile à lire. La sortie est représentée horizontalement sur plusieurs colonnes. PIVOT est similaire à une requête GROUP BY avec une agrégation, utilisant une expression agrégée pour spécifier un format de sortie. Toutefois, contrairement à GROUP BY, les résultats sont renvoyés sous forme de colonnes plutôt que de lignes.

Pour bénéficier d’exemples montrant comment interroger avec PIVOT et UNPIVOT, consultez Exemples PIVOT et UNPIVOT.

UNPIVOT

Rotation de colonnes en lignes avec UNPIVOT — L'opérateur transforme les colonnes de résultats d'une table d'entrée ou de résultats de requête en lignes, afin de faciliter la lecture de la sortie. UNPIVOT combine les données de ses colonnes d’entrée en deux colonnes de résultats : une colonne de noms et une colonne de valeurs. La colonne de noms contient les noms de colonnes provenant de l’entrée, sous forme d’entrées de ligne. La colonne de valeurs contient des valeurs provenant des colonnes d’entrée, telles que les résultats d’une agrégation. Par exemple, le nombre d’éléments dans différentes catégories.

Dépivotement de l'objet avec UNPIVOT (SUPER) : vous pouvez effectuer le dépivotement d'un objet, l'expression étant une expression SUPER faisant référence à un autre élément de la clause FROM. Pour de plus amples informations, veuillez consulter Dépivotement d'objet. Il contient également des exemples qui montrent comment interroger des données semi-structurées, telles que des données au format JSON.

Notes d'utilisation

Les colonnes de jointure doivent avoir des types de données comparables.

Une jointure NATURAL ou USING conserve seulement l'une de chaque paire de colonnes de jointure dans le jeu de résultats intermédiaire.

Une jointure avec la syntaxe ON conserve les deux colonnes de jointure dans son jeu de résultats intermédiaire.

Voir aussi Clause WITH.

Exemples PIVOT et UNPIVOT

PIVOT et UNPIVOT sont des paramètres de la clause FROM qui font pivoter la sortie de la requête des lignes vers les colonnes et des colonnes vers les lignes, respectivement. Ils représentent des résultats de requêtes tabulaires dans un format facile à lire. Les exemples suivants utilisent des données et des requêtes de test pour montrer comment les utiliser.

Pour plus d'informations sur ces paramètres et d'autres, consultez Clause FROM.

Exemples PIVOT

Configurez l'exemple de table et de données, puis utilisez-les pour exécuter les requêtes d'exemple suivantes.

```
CREATE TABLE part (
    partname varchar,
    manufacturer varchar,
    quality int,
    price decimal(12, 2)
);

INSERT INTO part VALUES ('prop', 'local parts co', 2, 10.00);
INSERT INTO part VALUES ('prop', 'big parts co', NULL, 9.00);
```

```
INSERT INTO part VALUES ('prop', 'small parts co', 1, 12.00);

INSERT INTO part VALUES ('rudder', 'local parts co', 1, 2.50);
INSERT INTO part VALUES ('rudder', 'big parts co', 2, 3.75);
INSERT INTO part VALUES ('rudder', 'small parts co', NULL, 1.90);

INSERT INTO part VALUES ('wing', 'local parts co', NULL, 7.50);
INSERT INTO part VALUES ('wing', 'big parts co', 1, 15.20);
INSERT INTO part VALUES ('wing', 'small parts co', NULL, 11.80);
```

PIVOT sur `partname` avec une agrégation `AVG` sur `price`.

```
SELECT *
FROM (SELECT partname, price FROM part) PIVOT (
    AVG(price) FOR partname IN ('prop', 'rudder', 'wing')
);
```

La requête génèrera la sortie suivante.

```
  prop   |  rudder  |  wing
---------+----------+---------
 10.33   | 2.71     | 11.50
```

Dans l'exemple précédent, les résultats sont transformés en colonnes. L'exemple suivant montre une requête `GROUP BY` qui renvoie les prix moyens sous forme de lignes plutôt que de colonnes.

```
SELECT partname, avg(price)
FROM (SELECT partname, price FROM part)
WHERE partname IN ('prop', 'rudder', 'wing')
GROUP BY partname;
```

La requête génèrera la sortie suivante.

```
 partname |  avg
----------+-------
 prop     | 10.33
 rudder   |  2.71
 wing     | 11.50
```

Un exemple `PIVOT` avec `manufacturer` en tant que colonne implicite.

```
SELECT *
FROM (SELECT quality, manufacturer FROM part) PIVOT (
    count(*) FOR quality IN (1, 2, NULL)
);
```

La requête génèrera la sortie suivante.

```
 manufacturer      | 1  | 2  | null
-------------------+----+----+------
 local parts co    | 1  | 1  |  1
 big parts co      | 1  | 1  |  1
 small parts co    | 1  | 0  |  2
```

Les colonnes de table d'entrée qui ne sont pas référencées dans la définition de `PIVOT` sont ajoutées implicitement à la table de résultats. C'est le cas pour la colonne `manufacturer` de l'exemple précédent. L'exemple montre également que `NULL` est une valeur valide pour l'opérateur `IN`.

`PIVOT` dans l'exemple ci-dessus renvoie des informations similaires à celles de la requête suivante, qui inclut `GROUP BY`. La différence est que `PIVOT` renvoie la valeur `0` pour la colonne `2` et le fabricant `small parts co`. La requête `GROUP BY` ne contient pas de ligne correspondante. Dans la plupart des cas, `PIVOT` insère `NULL` si une ligne ne contient pas de données d'entrée pour une colonne donnée. Toutefois, l'agrégat de nombre ne renvoie pas `NULL` et `0` est la valeur par défaut.

```
SELECT manufacturer, quality, count(*)
FROM (SELECT quality, manufacturer FROM part)
WHERE quality IN (1, 2) OR quality IS NULL
GROUP BY manufacturer, quality
ORDER BY manufacturer;
```

La requête génèrera la sortie suivante.

```
 manufacturer        | quality | count
---------------------+---------+-------
 big parts co        |         |     1
 big parts co        |       2 |     1
 big parts co        |       1 |     1
 local parts co      |       2 |     1
 local parts co      |       1 |     1
 local parts co      |         |     1
 small parts co      |       1 |     1
```

```
 small parts co      |         |     2
```

L'opérateur PIVOT accepte les alias facultatifs sur l'expression agrégée et sur chaque valeur pour l'opérateur `IN`. Utilisez des alias pour personnaliser les noms des colonnes. S'il n'y a pas d'alias agrégé, seuls les alias de liste `IN` sont utilisés. Sinon, l'alias agrégé est ajouté au nom de la colonne avec un trait de soulignement pour séparer les noms.

```
SELECT *
FROM (SELECT quality, manufacturer FROM part) PIVOT (
    count(*) AS count FOR quality IN (1 AS high, 2 AS low, NULL AS na)
);
```

La requête génèrera la sortie suivante.

```
 manufacturer      | high_count  | low_count | na_count
-------------------+-------------+-----------+----------
 local parts co    |           1 |         1 |        1
 big parts co      |           1 |         1 |        1
 small parts co    |           1 |         0 |        2
```

Configurez les exemples de table et de données suivants, puis utilisez-les pour exécuter les requêtes d'exemple suivantes. Les données représentent les dates de réservation pour un ensemble d'hôtels.

```
CREATE TABLE bookings (
    booking_id int,
    hotel_code char(8),
    booking_date date,
    price decimal(12, 2)
);

INSERT INTO bookings VALUES (1, 'FOREST_L', '02/01/2023', 75.12);
INSERT INTO bookings VALUES (2, 'FOREST_L', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (3, 'FOREST_L', '02/04/2023', 85.54);

INSERT INTO bookings VALUES (4, 'FOREST_L', '02/08/2023', 75.00);
INSERT INTO bookings VALUES (5, 'FOREST_L', '02/11/2023', 75.00);
INSERT INTO bookings VALUES (6, 'FOREST_L', '02/14/2023', 90.00);

INSERT INTO bookings VALUES (7, 'FOREST_L', '02/21/2023', 60.00);
INSERT INTO bookings VALUES (8, 'FOREST_L', '02/22/2023', 85.00);
INSERT INTO bookings VALUES (9, 'FOREST_L', '02/27/2023', 90.00);
```

```
INSERT INTO bookings VALUES (10, 'DESERT_S', '02/01/2023', 98.00);
INSERT INTO bookings VALUES (11, 'DESERT_S', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (12, 'DESERT_S', '02/04/2023', 85.00);

INSERT INTO bookings VALUES (13, 'DESERT_S', '02/05/2023', 75.00);
INSERT INTO bookings VALUES (14, 'DESERT_S', '02/06/2023', 34.00);
INSERT INTO bookings VALUES (15, 'DESERT_S', '02/09/2023', 85.00);

INSERT INTO bookings VALUES (16, 'DESERT_S', '02/12/2023', 23.00);
INSERT INTO bookings VALUES (17, 'DESERT_S', '02/13/2023', 76.00);
INSERT INTO bookings VALUES (18, 'DESERT_S', '02/14/2023', 85.00);

INSERT INTO bookings VALUES (19, 'OCEAN_WV', '02/01/2023', 98.00);
INSERT INTO bookings VALUES (20, 'OCEAN_WV', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (21, 'OCEAN_WV', '02/04/2023', 85.00);

INSERT INTO bookings VALUES (22, 'OCEAN_WV', '02/06/2023', 75.00);
INSERT INTO bookings VALUES (23, 'OCEAN_WV', '02/09/2023', 34.00);
INSERT INTO bookings VALUES (24, 'OCEAN_WV', '02/12/2023', 85.00);

INSERT INTO bookings VALUES (25, 'OCEAN_WV', '02/13/2023', 23.00);
INSERT INTO bookings VALUES (26, 'OCEAN_WV', '02/14/2023', 76.00);
INSERT INTO bookings VALUES (27, 'OCEAN_WV', '02/16/2023', 85.00);

INSERT INTO bookings VALUES (28, 'CITY_BLD', '02/01/2023', 98.00);
INSERT INTO bookings VALUES (29, 'CITY_BLD', '02/02/2023', 75.00);
INSERT INTO bookings VALUES (30, 'CITY_BLD', '02/04/2023', 85.00);

INSERT INTO bookings VALUES (31, 'CITY_BLD', '02/12/2023', 75.00);
INSERT INTO bookings VALUES (32, 'CITY_BLD', '02/13/2023', 34.00);
INSERT INTO bookings VALUES (33, 'CITY_BLD', '02/17/2023', 85.00);

INSERT INTO bookings VALUES (34, 'CITY_BLD', '02/22/2023', 23.00);
INSERT INTO bookings VALUES (35, 'CITY_BLD', '02/23/2023', 76.00);
INSERT INTO bookings VALUES (36, 'CITY_BLD', '02/24/2023', 85.00);
```

Dans cet exemple de requête, les enregistrements de réservations sont comptabilisés pour donner un total pour chaque semaine. La date de fin de chaque semaine devient un nom de colonne.

```
SELECT * FROM
    (SELECT
       booking_id,
```

```
       (date_trunc('week', booking_date::date) + '5 days'::interval)::date as enddate,
       hotel_code AS "hotel code"
FROM bookings
) PIVOT (
    count(booking_id) FOR enddate IN ('2023-02-04','2023-02-11','2023-02-18')
);
```

La requête génèrera la sortie suivante.

```
 hotel code | 2023-02-04  | 2023-02-11 | 2023-02-18
------------+-------------+------------+----------
 FOREST_L   |           3 |          2 |        1
 DESERT_S   |           4 |          3 |        2
 OCEAN_WV   |           3 |          3 |        3
 CITY_BLD   |           3 |          1 |        2
```

Amazon Redshift ne prend pas en charge CROSSTAB pour pivoter sur plusieurs colonnes. Mais vous pouvez transformer les données de ligne en colonnes, comme pour un regroupement avec PIVOT, à l'aide d'une requête telle que la suivante. L'exemple précédent utilise les mêmes données de réservation que celles de l'exemple précédent.

```
SELECT
  booking_date,
  MAX(CASE WHEN hotel_code = 'FOREST_L' THEN 'forest is booked' ELSE '' END) AS
 FOREST_L,
  MAX(CASE WHEN hotel_code = 'DESERT_S' THEN 'desert is booked' ELSE '' END) AS
 DESERT_S,
  MAX(CASE WHEN hotel_code = 'OCEAN_WV' THEN 'ocean is booked' ELSE '' END)  AS
 OCEAN_WV
FROM bookings
GROUP BY booking_date
ORDER BY booking_date asc;
```

L'exemple de requête donne lieu à des dates de réservation répertoriées à côté de phrases courtes qui indiquent quels hôtels sont réservés.

```
 booking_date  | forest_l         | desert_s         | ocean_wv
---------------+------------------+------------------+--------------------
 2023-02-01    | forest is booked | desert is booked |  ocean is booked
 2023-02-02    | forest is booked | desert is booked |  ocean is booked
 2023-02-04    | forest is booked | desert is booked |  ocean is booked
```

```
 2023-02-05    |                 | desert is booked |
 2023-02-06    |                 | desert is booked |
```

Voici des notes d'utilisation pour PIVOT :

- PIVOTpeut être appliqué aux tables, aux sous-requêtes et aux expressions de table communes (CTEs). PIVOTne peut être appliqué à aucune JOIN expression, récursive CTEs ou UNPIVOT expression. PIVOT De plus, les expressions non imbriquées SUPER et les tables imbriquées Redshift Spectrum ne sont pas prises en charge.
- PIVOT prend en charge les fonctions agrégées COUNT, SUM, MIN, MAX et AVG.
- L'expression agrégée PIVOT doit être un appel d'une fonction agrégée prise en charge. Les expressions complexes en plus de l'agrégat ne sont pas prises en charge. Les arguments agrégés ne peuvent pas contenir de références à d'autres tables que la table d'entrée PIVOT. Les références corrélées à une requête parente ne sont pas prises en charge. L'argument agrégé peut contenir des sous-requêtes. Elles peuvent être corrélées en interne ou sur la table d'entrée PIVOT.
- Les valeurs de liste PIVOT IN ne peuvent pas être des références de colonnes ou des sous-requêtes. Chaque valeur doit être de type compatible avec la référence de colonne FOR.
- Si les valeurs de liste IN n'ont pas d'alias, PIVOT génère des noms de colonnes par défaut. Pour des valeurs constantes IN telles que « abc » ou 5, le nom de colonne par défaut est la constante elle-même. Pour toute expression complexe, le nom de la colonne est un nom par défaut Amazon Redshift standard tel que ?column?.

Exemples UNPIVOT

Configurez les exemples de données et utilisez-les pour exécuter les exemples suivants.

```
CREATE TABLE count_by_color (quality varchar, red int, green int, blue int);

INSERT INTO count_by_color VALUES ('high', 15, 20, 7);
INSERT INTO count_by_color VALUES ('normal', 35, NULL, 40);
INSERT INTO count_by_color VALUES ('low', 10, 23, NULL);
```

UNPIVOT sur les colonnes d'entrée rouges, vertes et bleues.

```
SELECT *
FROM (SELECT red, green, blue FROM count_by_color) UNPIVOT (
    cnt FOR color IN (red, green, blue)
);
```

La requête génèrera la sortie suivante.

```
 color | cnt
-------+-----
 red   |  15
 red   |  35
 red   |  10
 green |  20
 green |  23
 blue  |   7
 blue  |  40
```

Par défaut, les valeurs NULL de la colonne d'entrée sont ignorées et ne produisent pas de ligne de résultats.

L'exemple suivant montre UNPIVOT avec INCLUDE NULLS.

```
SELECT *
FROM (
    SELECT red, green, blue
    FROM count_by_color
) UNPIVOT INCLUDE NULLS (
    cnt FOR color IN (red, green, blue)
);
```

En voici le résultat obtenu.

```
 color | cnt
-------+-----
 red   |  15
 red   |  35
 red   |  10
 green |  20
 green |
 green |  23
 blue  |   7
 blue  |  40
 blue  |
```

Si le paramètre INCLUDING NULLS est défini, les valeurs d'entrée NULL génèrent des lignes de résultats.

The following query shows UNPIVOT avec quality en tant que colonne implicite.

```
SELECT *
FROM count_by_color UNPIVOT (
    cnt FOR color IN (red, green, blue)
);
```

La requête génèrera la sortie suivante.

```
 quality | color | cnt
---------+-------+-----
 high    | red   |  15
 normal  | red   |  35
 low     | red   |  10
 high    | green |  20
 low     | green |  23
 high    | blue  |   7
 normal  | blue  |  40
```

Les colonnes de la table d'entrée qui ne sont pas référencées dans la définition de UNPIVOT sont ajoutées implicitement à la table de résultats. Dans cet exemple, c'est le cas pour la colonne quality.

L'exemple suivant en est une illustration de UNPIVOT avec des alias pour les valeurs dans la liste IN.

```
SELECT *
FROM count_by_color UNPIVOT (
    cnt FOR color IN (red AS r, green AS g, blue AS b)
);
```

La requête précédente génère le résultat suivant.

```
 quality | color | cnt
---------+-------+-----
 high    | r     |  15
 normal  | r     |  35
 low     | r     |  10
 high    | g     |  20
 low     | g     |  23
 high    | b     |   7
 normal  | b     |  40
```

L'opérateur UNPIVOT accepte les alias facultatifs sur chaque valeur de liste IN. Chaque alias permet de personnaliser les données de chaque colonne value.

Voici des notes d'utilisation pour UNPIVOT.

- UNPIVOTpeut être appliqué aux tables, aux sous-requêtes et aux expressions de table communes (CTEs). UNPIVOTne peut être appliqué à aucune JOIN expression, récursive CTEs ou UNPIVOT expression. PIVOT De plus, les expressions non imbriquées SUPER et les tables imbriquées Redshift Spectrum ne sont pas prises en charge.
- La liste UNPIVOT IN doit contenir uniquement des références de colonnes de table d'entrée. Les colonnes de la liste IN doivent avoir un type commun avec lequel elles sont toutes compatibles. La colonne de valeurs UNPIVOT a ce type commun. La colonne de noms UNPIVOT est de type VARCHAR.
- Si une valeur de liste IN ne possède pas d'alias, UNPIVOT utilise le nom de la colonne comme valeur par défaut.

Exemples de clause JOIN

Une clause SQL JOIN permet de combiner les données de deux ou plusieurs tables sur la base de champs communs. Les résultats peuvent ou non changer en fonction de la méthode de jointure spécifiée. Pour obtenir plus d'informations sur la syntaxe d'une clause JOIN, consultez Paramètres.

Les exemples suivants utilisent les exemples de données TICKIT. Pour obtenir plus d'informations sur le schéma de la base de données, consultez Exemple de base de données. Pour savoir comment charger des exemples de données, consultez la section Chargement de données dans le guide de démarrage Amazon Redshift.

La requête suivante est une jointure interne (sans le mot-clé JOIN) entre la table LISTING et la table SALES, où la valeur LISTID de la table LISTING est comprise entre 1 et 5. Cette requête met en correspondance les valeurs de la colonne LISTID dans les tables LISTING (table de gauche) et SALES (table de droite). Les résultats montrent que les valeurs LISTID 1, 4 et 5 correspondent aux critères.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing, sales
where listing.listid = sales.listid
and listing.listid between 1 and 5
group by 1
```

```
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

La requête suivante est une jointure externe gauche. Les jointures externes gauche et droite conservent les valeurs de l'une des tables jointes quand aucune correspondance n'est trouvée dans l'autre table. Les tables gauche et droite sont la première et la deuxième répertoriées dans la syntaxe. Les valeurs NULL sont utilisées pour combler les « écarts » du jeu de résultats. Cette requête fait correspondre les valeurs de la colonne LISTID dans la table LISTING (la table de gauche) et la table SALES (la table de droite). Les résultats montrent que LISTIDs 2 et 3 n'ont donné lieu à aucune vente.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing left outer join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     2 | NULL   | NULL
     3 | NULL   | NULL
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

La requête suivante est une jointure externe droite. Cette requête fait correspondre les valeurs de la colonne LISTID dans la table LISTING (la table de gauche) et la table SALES (la table de droite). Les résultats montrent que LISTIDs 1, 4 et 5 correspondent aux critères.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing right outer join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
```

```
-------+--------+--------
     1 | 728.00 | 109.20
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

La requête suivante est une jointure complète. Les jointures complètes conservent les valeurs des tables jointes lorsqu'aucune correspondance n'est trouvée dans l'autre table. Les tables gauche et droite sont la première et la deuxième répertoriées dans la syntaxe. Les valeurs NULL sont utilisées pour combler les « écarts » du jeu de résultats. Cette requête fait correspondre les valeurs de la colonne LISTID dans la table LISTING (la table de gauche) et la table SALES (la table de droite). Les résultats montrent que LISTIDs 2 et 3 n'ont donné lieu à aucune vente.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing full join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     2 | NULL   | NULL
     3 | NULL   | NULL
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

La requête suivante est une jointure complète. Cette requête fait correspondre les valeurs de la colonne LISTID dans la table LISTING (la table de gauche) et la table SALES (la table de droite). Seules les lignes qui ne génèrent aucune vente (LISTIDs 2 et 3) apparaissent dans les résultats.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from listing full join sales on sales.listid = listing.listid
where listing.listid between 1 and 5
and (listing.listid IS NULL or sales.listid IS NULL)
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     2 | NULL   | NULL
     3 | NULL   | NULL
```

L'exemple suivant est une jointure interne avec la clause ON. Dans ce cas, les lignes NULL ne sont pas renvoyées.

```
select listing.listid, sum(pricepaid) as price, sum(commission) as comm
from sales join listing
on sales.listid=listing.listid and sales.eventid=listing.eventid
where listing.listid between 1 and 5
group by 1
order by 1;

listid | price  |  comm
-------+--------+--------
     1 | 728.00 | 109.20
     4 |  76.00 |  11.40
     5 | 525.00 |  78.75
```

La requête suivante est une jointure croisée ou cartésienne de la table LISTING et de la table SALES avec un prédicat pour limiter les résultats. Cette requête correspond aux valeurs des colonnes LISTID de la table SALES et de la table LISTING pour LISTIDs 1, 2, 3, 4 et 5 dans les deux tables. Les résultats montrent que 20 lignes correspondent aux critères.

```
select sales.listid as sales_listid, listing.listid as listing_listid
from sales cross join listing
where sales.listid between 1 and 5
and listing.listid between 1 and 5
order by 1,2;

sales_listid | listing_listid
-------------+---------------
1            | 1
1            | 2
1            | 3
1            | 4
1            | 5
4            | 1
4            | 2
4            | 3
4            | 4
4            | 5
5            | 1
5            | 1
5            | 2
```

```
5            | 2
5            | 3
5            | 3
5            | 4
5            | 4
5            | 5
5            | 5
```

L'exemple suivant est une jointure naturelle entre deux tables. Dans ce cas, les colonnes listid, sellerid, eventid et dateid présentent des noms et des types de données identiques dans les deux tables et sont donc utilisées comme colonnes de jointure. Les résultats sont limités à seulement cinq lignes.

```
select listid, sellerid, eventid, dateid, numtickets
from listing natural join sales
order by 1
limit 5;

listid | sellerid  | eventid | dateid | numtickets
-------+-----------+---------+--------+-----------
113    | 29704     | 4699    | 2075   | 22
115    | 39115     | 3513    | 2062   | 14
116    | 43314     | 8675    | 1910   | 28
118    | 6079      | 1611    | 1862   | 9
163    | 24880     | 8253    | 1888   | 14
```

L'exemple suivant est une jointure entre deux tables avec la clause USING. Dans ce cas, les colonnes listid et eventid sont utilisées comme colonnes de jointure. Les résultats sont limités à seulement cinq lignes.

```
select listid, listing.sellerid, eventid, listing.dateid, numtickets
from listing join sales
using (listid, eventid)
order by 1
limit 5;

listid | sellerid | eventid | dateid | numtickets
-------+----------+---------+--------+-----------
1      | 36861    | 7872    | 1850   | 10
4      | 8117     | 4337    | 1970   | 8
5      | 1616     | 8647    | 1963   | 4
5      | 1616     | 8647    | 1963   | 4
```

```
6      | 47402    | 8240    | 2053   | 18
```

La requête suivante est une jointure interne de deux sous-requêtes de la clause FROM. La requête recherche le nombre de billets vendus et invendus pour les différentes catégories d'événements (concerts et spectacles). Les sous-requêtes de la clause FROM sont des sous-requêtes de table ; elles peuvent renvoyer plusieurs lignes et colonnes.

```
select catgroup1, sold, unsold
from
(select catgroup, sum(qtysold) as sold
from category c, event e, sales s
where c.catid = e.catid and e.eventid = s.eventid
group by catgroup) as a(catgroup1, sold)
join
(select catgroup, sum(numtickets)-sum(qtysold) as unsold
from category c, event e, sales s, listing l
where c.catid = e.catid and e.eventid = s.eventid
and s.listid = l.listid
group by catgroup) as b(catgroup2, unsold)

on a.catgroup1 = b.catgroup2
order by 1;

catgroup1 |  sold  | unsold
----------+--------+--------
Concerts  | 195444 |1067199
Shows     | 149905 | 817736
```

Clause WHERE

La clause WHERE contient les conditions qui joignent les tables ou appliquent les prédicats aux colonnes des tables. Les tables peuvent être à jointure interne en utilisant la syntaxe appropriée dans la clause WHERE ou FROM. Les critères de jointure externe doivent être spécifiés dans la clause FROM.

Syntaxe

```
[ WHERE condition ]
```

condition

Toute condition avec un résultat Boolean, comme une condition de jointure ou un prédicat sur une colonne de table. Les exemples suivants sont des conditions de jointure valides :

```
sales.listid=listing.listid
sales.listid<>listing.listid
```

Les exemples suivants sont des conditions valides sur les colonnes des tables :

```
catgroup like 'S%'
venueseats between 20000 and 50000
eventname in('Jersey Boys','Spamalot')
year=2008
length(catdesc)>25
date_part(month, caldate)=6
```

Les conditions peuvent être simples ou complexes ; pour les conditions complexes, vous pouvez utiliser des parenthèses afin d'isoler des unités logiques. Dans l'exemple suivant, la condition de jointure est placée entre parenthèses.

```
where (category.catid=event.catid) and category.catid in(6,7,8)
```

Notes d'utilisation

Vous pouvez utiliser des alias dans la clause WHERE pour référencer les expressions de liste de sélection.

Vous ne pouvez pas limiter les résultats des fonctions d'agrégation dans la clause WHERE ; utilisez à cette fin la clause HAVING.

Les colonnes qui sont limités dans la clause WHERE doivent provenir de références de table de la clause FROM.

exemple

La requête suivante utilise une combinaison de différentes restrictions de clause WHERE, y compris une condition de jointure pour les tables SALES et EVENT, un prédicat sur la colonne EVENTNAME et deux prédicats sur la colonne STARTTIME.

```
select eventname, starttime, pricepaid/qtysold as costperticket, qtysold
from sales, event
where sales.eventid = event.eventid
and eventname='Hannah Montana'
and date_part(quarter, starttime) in(1,2)
and date_part(year, starttime) = 2008
order by 3 desc, 4, 2, 1 limit 10;

eventname    |      starttime      |   costperticket   | qtysold
---------------+---------------------+-------------------+---------
Hannah Montana | 2008-06-07 14:00:00 |     1706.00000000 |       2
Hannah Montana | 2008-05-01 19:00:00 |     1658.00000000 |       2
Hannah Montana | 2008-06-07 14:00:00 |     1479.00000000 |       1
Hannah Montana | 2008-06-07 14:00:00 |     1479.00000000 |       3
Hannah Montana | 2008-06-07 14:00:00 |     1163.00000000 |       1
Hannah Montana | 2008-06-07 14:00:00 |     1163.00000000 |       2
Hannah Montana | 2008-06-07 14:00:00 |     1163.00000000 |       4
Hannah Montana | 2008-05-01 19:00:00 |      497.00000000 |       1
Hannah Montana | 2008-05-01 19:00:00 |      497.00000000 |       2
Hannah Montana | 2008-05-01 19:00:00 |      497.00000000 |       4
(10 rows)
```

Jointures externes Oracle dans la clause WHERE

Pour des raisons de compatibilité Oracle, Amazon Redshift prend en charge l'opérateur de jointure externe Oracle (+) dans les conditions de clause WHERE. Cet opérateur doit être utilisé uniquement dans la définition de conditions de jointure externe ; n'essayez pas de l'utiliser dans d'autres contextes. Les autres utilisations de cet opérateur sont automatiquement ignorées dans la plupart des cas.

Une jointure externe renvoie toutes les lignes que la jointure interne équivalente renvoie, plus les lignes non correspondantes d'une ou de deux tables. Dans la clause FROM, vous pouvez spécifier des jointures externes gauches, droites et complètes. Dans la clause WHERE, vous ne pouvez spécifier que des jointures externes gauches et droites.

Pour les tables de jointure externes TABLE1 TABLE2 et pour renvoyer des lignes non correspondantes depuis TABLE1 (une jointure externe gauche), spécifiez-le `TABLE1 LEFT OUTER JOIN TABLE2` dans la clause FROM ou appliquez l'opérateur (+) à toutes les colonnes de jointure depuis TABLE2 la clause WHERE. Pour toutes les lignes TABLE1 qui ne contiennent aucune ligne correspondante TABLE2, le résultat de la requête contient des valeurs nulles pour toutes les expressions de liste de sélection contenant des colonnes provenant de TABLE2.

Pour produire le même comportement pour toutes les lignes TABLE2 qui ne contiennent aucune ligne correspondante TABLE1, spécifiez-le `TABLE1 RIGHT OUTER JOIN TABLE2` dans la clause FROM ou appliquez l'opérateur (+) à toutes les colonnes jointes depuis TABLE1 la clause WHERE.

Syntaxe de base

```
[ WHERE {
[ table1.column1 = table2.column1(+) ]
[ table1.column1(+) = table2.column1 ]
}
```

La première condition est équivalente à :

```
from table1 left outer join table2
on table1.column1=table2.column1
```

La deuxième condition est équivalente à :

```
from table1 right outer join table2
on table1.column1=table2.column1
```

> **Note**
>
> La syntaxe présentée ici couvre le cas simple d'une équijointure sur une paire de colonnes de jointure. Cependant, d'autres types de conditions de comparaison et plusieurs paires de colonnes de jointure sont également valides.

Par exemple, la clause WHERE suivante définit une jointure externe sur deux paires de colonnes. L'opérateur (+) doit être associé à la même table dans les deux conditions :

```
where table1.col1 > table2.col1(+)
and table1.col2 = table2.col2(+)
```

Notes d'utilisation

Si possible, utilisez la syntaxe OUTER JOIN de la clause standard FROM au lieu de l'opérateur (+) dans la clause WHERE. Les requêtes qui contiennent l'opérateur (+) sont soumises aux règles suivantes :

- Vous ne pouvez utiliser que l'opérateur (+) dans la clause WHERE, et uniquement en référence aux colonnes des tables ou des vues.
- Vous ne pouvez pas appliquer l'opérateur (+) aux expressions. Cependant, une expression peut contenir des colonnes qui utilisent l'opérateur (+). Par exemple, la condition de jointure suivante renvoie une erreur de syntaxe :

  ```
  event.eventid*10(+)=category.catid
  ```

  Cependant, la condition de jointure suivante est valide :

  ```
  event.eventid(+)*10=category.catid
  ```

- Vous ne pouvez pas utiliser l'opérateur (+) dans un bloc de requête qui contient également la syntaxe de jointure de la clause FROM.
- Si deux tables sont jointes sur plusieurs conditions de jointure, vous devez utiliser l'opérateur (+) dans la totalité de ces conditions ou dans aucune d'entre elles. Une jointure avec des styles de syntaxe mixtes est exécutée comme jointure interne, sans avertissement.
- L'opérateur (+) ne génère pas une jointure externe si vous joignez une table de la requête externe à une table qui résulte d'une requête interne.
- Pour utiliser l'opérateur (+) et créer une jointure externe d'une table avec elle-même, vous devez définir les alias de table dans la clause FROM et les référencer dans la condition de jointure :

  ```
  select count(*)
  from event a, event b
  where a.eventid(+)=b.catid;

  count
  -------
  8798
  (1 row)
  ```

- Vous ne pouvez pas associer une condition de jointure contenant l'opérateur (+) avec une condition OR ou une condition IN. Par exemple :

  ```
  select count(*) from sales, listing
  where sales.listid(+)=listing.listid or sales.salesid=0;
  ERROR:  Outer join operator (+) not allowed in operand of OR or IN.
  ```

- Dans une clause WHERE qui crée une jointure externe avec plus de deux tables, l'opérateur (+) ne peut être appliqué qu'une seule fois à une table donnée. Dans l'exemple suivant, la table SALES ne peut pas être référencée par l'opérateur (+) dans deux jointures successives.

```
select count(*) from sales, listing, event
where sales.listid(+)=listing.listid and sales.dateid(+)=date.dateid;
ERROR:  A table may be outer joined to at most one other table.
```

- Si la condition de jointure externe de la clause WHERE compare une colonne à TABLE2 une constante, appliquez l'opérateur (+) à la colonne. Si vous n'incluez pas l'opérateur, les lignes jointes à l'extérieur de TABLE1, qui contiennent des valeurs nulles pour la colonne restreinte, sont éliminées. Consultez la section Exemples ci-dessous.

Exemples

La requête de jointure suivante spécifie une jointure externe gauche des tables SALES et LISTING, sur leurs colonnes LISTID :

```
select count(*)
from sales, listing
where sales.listid = listing.listid(+);

count
--------
172456
(1 row)
```

La requête équivalente suivante produit le même résultat, mais utilise la syntaxe de jointure de la clause FROM :

```
select count(*)
from sales left outer join listing on sales.listid = listing.listid;

count
--------
172456
(1 row)
```

La table SALES ne contient pas d'enregistrements pour toutes les listes de la table LISTING, car certaines listes ne se traduisent pas par des ventes. La requête suivante crée une jointure externe de

SALES et de LISTING, et renvoie les lignes de LISTING même lorsque la table SALES ne rapporte aucune vente pour un ID de liste donné. Les colonnes PRICE et COMM, dérivées de la table SALES, contiennent des valeurs null dans le jeu de résultats pour ces lignes sans correspondance.

```
select listing.listid, sum(pricepaid) as price,
sum(commission) as comm
from listing, sales
where sales.listid(+) = listing.listid and listing.listid between 1 and 5
group by 1 order by 1;

listid | price  |  comm
--------+--------+--------
1 | 728.00 | 109.20
2 |        |
3 |        |
4 |  76.00 |  11.40
5 | 525.00 |  78.75
(5 rows)
```

Notez que lorsque l'opérateur de jointure de la clause WHERE est utilisé, l'ordre des tables dans la clause FROM n'importe pas.

L'exemple d'une condition de jointure externe plus complexe dans la clause WHERE est celui où la condition se compose d'une comparaison entre deux Colonnes de la table et d'une comparaison avec une constante :

```
where category.catid=event.catid(+) and eventid(+)=796;
```

Notez que l'opérateur (+) est utilisé à deux emplacements : d'abord dans la comparaison d'égalité entre les tables et ensuite dans la condition de comparaison pour la colonne EVENTID. Le résultat de cette syntaxe est la conservation des lignes de jointure externe lors de l'évaluation de la restriction sur EVENTID. Si vous supprimez l'opérateur (+) de la restriction EVENTID, la requête traite cette restriction comme filtre, pas dans le cadre de la condition de jointure externe. A leur tour, les lignes de jointure externe qui contiennent des valeurs null pour EVENTID sont éliminées du jeu de résultats.

Voici une requête complète qui illustre ce comportement :

```
select catname, catgroup, eventid
from category, event
where category.catid=event.catid(+) and eventid(+)=796;
```

```
catname | catgroup | eventid
-----------+----------+---------
Classical | Concerts |
Jazz | Concerts |
MLB | Sports   |
MLS | Sports   |
Musicals | Shows    | 796
NBA | Sports   |
NFL | Sports   |
NHL | Sports   |
Opera | Shows    |
Plays | Shows    |
Pop | Concerts |
(11 rows)
```

La requête équivalente à l'aide de la syntaxe de la clause FROM est la suivante :

```
select catname, catgroup, eventid
from category left join event
on category.catid=event.catid and eventid=796;
```

Si vous supprimez le deuxième opérateur (+) de la version de la clause WHERE de cette requête, elle renvoie uniquement 1 ligne (celle où `eventid=796`).

```
select catname, catgroup, eventid
from category, event
where category.catid=event.catid(+) and eventid=796;

catname | catgroup | eventid
-----------+----------+---------
Musicals | Shows    | 796
(1 row)
```

Clause GROUP BY

La clause GROUP BY identifie les colonnes de regroupement de la requête. Il est utilisé pour regrouper les lignes d'un tableau qui ont les mêmes valeurs dans toutes les colonnes répertoriées. L'ordre dans lequel les colonnes sont répertoriées n'a pas d'importance. Le résultat est de combiner chaque ensemble de lignes ayant des valeurs communes en une seule ligne de groupe qui représente toutes les lignes du groupe. Utilisez un GROUP BY pour éliminer la redondance dans la sortie et pour calculer les agrégats qui s'appliquent aux groupes. Les colonnes de regroupement

doivent être déclarées lorsque la requête calcule les regroupements avec des fonctions standard telles que SUM, AVG et COUNT. Pour de plus amples informations, veuillez consulter Fonctions d'agrégation.

Syntaxe

```
[ GROUP BY  expression [, ...] | ALL | aggregation_extension  ]
```

où aggregation_extension est l'une des valeurs suivantes :

```
GROUPING SETS ( () | aggregation_extension [, ...] ) |
ROLLUP ( expr [, ...] ) |
CUBE ( expr [, ...] )
```

Paramètres

expression

La liste des colonnes ou des expressions doit correspondre à la liste des expressions non agrégées de la liste de sélection de la requête. Par exemple, imaginons la requête simple suivante.

```
select listid, eventid, sum(pricepaid) as revenue,
count(qtysold) as numtix
from sales
group by listid, eventid
order by 3, 4, 2, 1
limit 5;

listid | eventid | revenue | numtix
-------+---------+---------+--------
89397  |      47 |   20.00 |      1
106590 |      76 |   20.00 |      1
124683 |     393 |   20.00 |      1
103037 |     403 |   20.00 |      1
147685 |     429 |   20.00 |      1
(5 rows)
```

Dans cette requête, la liste de sélection se compose de deux expressions d'agrégation. La première utilise la fonction SUM et la seconde la fonction COUNT. Les deux autres colonnes, LISTID et EVENTID, doivent être déclarées en tant que colonnes de regroupement.

Les expressions de la clause GROUP BY peuvent également faire référence à la liste de sélection en utilisant des nombres ordinaux. Par exemple, l'exemple précédent peut être abrégé comme suit.

```
select listid, eventid, sum(pricepaid) as revenue,
count(qtysold) as numtix
from sales
group by 1,2
order by 3, 4, 2, 1
limit 5;

listid | eventid | revenue | numtix
-------+---------+---------+--------
89397  |      47 |   20.00 |      1
106590 |      76 |   20.00 |      1
124683 |     393 |   20.00 |      1
103037 |     403 |   20.00 |      1
147685 |     429 |   20.00 |      1
(5 rows)
```

ALL

ALL indique de regrouper en fonction de toutes les colonnes spécifiées dans la liste SELECT, à l'exception de celles qui sont agrégées. Par exemple, considérez la requête suivante qui regroupe par `col1` et `col2` sans avoir à les spécifier individuellement dans la clause GROUP BY. La colonne `col3` est l'argument de la `SUM` fonction et n'est donc pas groupée.

```
SELECT col1, col2 sum(col3) FROM testtable GROUP BY ALL
```

Si vous EXCLUEZ une colonne de la liste SELECT, la clause GROUP BY ALL ne regroupe pas les résultats en fonction de cette colonne spécifique.

```
SELECT * EXCLUDE col3 FROM testtable GROUP BY ALL
```

extension_d'agrégation

Vous pouvez utiliser les extensions d'agrégation GROUPING SETS, ROLLUP et CUBE pour effectuer plusieurs opérations GROUP BY dans une seule instruction. Pour plus d'informations sur les extensions d'agrégation et les fonctions associées, consultez Extensions de regroupement.

Exemples

Les exemples suivants utilisent la table SALES qui contient des colonnes : salesid, listid, sellerid, buyerid, eventid, dateid, qtysold, pricepaid, commission et saletime. Pour plus d'informations sur le tableau SALES, consultezExemple de base de données.

L'exemple suivant donne des groupes de requêtes par `salesid` et `listid` sans qu'il soit nécessaire de les spécifier individuellement dans la clause GROUP BY. La colonne `qtysold` est l'argument de la SUM fonction et n'est donc pas groupée.

```
SELECT salesid, listid, sum(qtysold) FROM sales GROUP BY ALL;

salesid | listid  | sum
--------+---------+------
33095   | 36572   | 2
88268   | 100813  | 4
110917  | 127048  | 1
...
```

L'exemple de requête suivant exclut plusieurs colonnes de la liste SELECT. GROUP BY ALL ne regroupe donc que salesid et listid.

```
SELECT * EXCLUDE sellerid, buyerid, eventid, dateid, qtysold, pricepaid, commission,
 saletime
FROM sales GROUP BY ALL;

salesid | listid
--------+---------
33095   | 36572
88268   | 100813
110917  | 127048
...
```

Extensions de regroupement

Amazon Redshift prend en charge les extensions d’agrégation permettant d’effectuer plusieurs opérations GROUP BY dans une seule instruction.

Les exemples d’extensions d’agrégation utilisent la table `orders`, qui contient les données de vente d’une entreprise d’électronique. Pour créer des `orders`, procédez comme suit.

```
CREATE TABLE ORDERS (
    ID INT,
    PRODUCT CHAR(20),
    CATEGORY CHAR(20),
    PRE_OWNED CHAR(1),
    COST DECIMAL
);

INSERT INTO ORDERS VALUES
    (0, 'laptop',       'computers',    'T', 1000),
    (1, 'smartphone',   'cellphones',   'T', 800),
    (2, 'smartphone',   'cellphones',   'T', 810),
    (3, 'laptop',       'computers',    'F', 1050),
    (4, 'mouse',        'computers',    'F', 50);
```

GROUPING SETS

Calcule un ou plusieurs jeux de regroupement dans une seule instruction. Un jeu de regroupement est l'ensemble d'une clause GROUP BY unique, un jeu de 0 colonne ou plus avec lequel vous pouvez regrouper le jeu de résultats d'une requête. GROUP BY GROUPING SETS revient à exécuter une requête UNION ALL sur un jeu de résultats groupé par différentes colonnes. Par exemple, GROUP BY GROUPING SETS((a), (b)) est équivalent à GROUP BY a UNION ALL GROUP BY b.

L'exemple suivant renvoie le coût des produits de la table des commandes, regroupés par catégories de produits et type de produits vendus.

```
SELECT category, product, sum(cost) as total
FROM orders
GROUP BY GROUPING SETS(category, product);

       category       |       product        | total
----------------------+----------------------+-------
 computers            |                      |  2100
 cellphones           |                      |  1610
                      | laptop               |  2050
                      | smartphone           |  1610
                      | mouse                |    50

(5 rows)
```

ROLLUP

Suppose une hiérarchie dans laquelle les colonnes précédentes sont considérées comme les parents des colonnes suivantes. ROLLUP regroupe les données par colonnes fournies et renvoie des lignes de sous-totaux supplémentaires représentant les totaux à tous les niveaux de colonnes de regroupement, en plus des lignes groupées. Par exemple, vous pouvez utiliser GROUP BY ROLLUP((a), (b)) pour renvoyer un jeu de résultats regroupé d'abord par a, puis par b en supposant que b est une sous-section de a. ROLLUP renvoie également une ligne contenant le jeu des résultats sans regrouper les colonnes.

GROUP BY ROLLUP((a), (b)) équivaut à GROUP BY GROUPING SETS((a,b), (a), ()).

L'exemple suivant renvoie le coût des produits de la table des commandes, regroupés d'abord par catégorie, puis par produit, le produit étant une subdivision de la catégorie.

```
SELECT category, product, sum(cost) as total
FROM orders
GROUP BY ROLLUP(category, product) ORDER BY 1,2;

       category       |       product        | total
----------------------+----------------------+-------
 cellphones           | smartphone           |  1610
 cellphones           |                      |  1610
 computers            | laptop               |  2050
 computers            | mouse                |    50
 computers            |                      |  2100
                      |                      |  3710
(6 rows)
```

CUBE

Regroupe les données par colonnes fournies et renvoie des lignes de sous-totaux supplémentaires représentant les totaux à tous les niveaux de colonnes de regroupement, en plus des lignes groupées. CUBE renvoie les mêmes lignes que ROLLUP, mais ajoute des lignes de sous-total supplémentaires pour chaque combinaison de colonnes de regroupement non couverte par ROLLUP. Par exemple, vous pouvez utiliserGROUP BY CUBE ((a), (b)) pour renvoyer un jeu de résultats regroupé d'abord par a, puis par b en supposant que b est une sous-section de a, puis par b uniquement. CUBE renvoie également une ligne contenant le jeu des résultats sans regrouper les colonnes.

GROUP BY CUBE((a), (b)) équivaut à GROUP BY GROUPING SETS((a, b), (a), (b), ()).

L'exemple suivant renvoie le coût des produits de la table des commandes, regroupés d'abord par catégorie, puis par produit, le produit étant une subdivision de la catégorie. Contrairement à l'exemple précédent pour ROLLUP, l'instruction renvoie des résultats pour chaque combinaison de colonnes de regroupement.

```
SELECT category, product, sum(cost) as total
FROM orders
GROUP BY CUBE(category, product) ORDER BY 1,2;

       category       |       product        | total
----------------------+----------------------+-------
 cellphones           | smartphone           |  1610
 cellphones           |                      |  1610
 computers            | laptop               |  2050
 computers            | mouse                |    50
 computers            |                      |  2100
                      | laptop               |  2050
                      | mouse                |    50
                      | smartphone           |  1610
                      |                      |  3710
(9 rows)
```

Fonctions GROUPING/GROUPING_ID

ROLLUP et CUBE ajoutent des valeurs NULL au jeu de résultats pour indiquer le sous-total des lignes. Par exemple, GROUP BY ROLLUP((a), (b)) renvoie une ou plusieurs lignes dont la valeur est NULL dans la colonne de regroupement b pour indiquer qu'il s'agit de sous-totaux des champs de la colonne de regroupement a. Ces valeurs NULL ne servent qu'à satisfaire le format des tuples renvoyés.

Lorsque vous exécutez des opérations GROUP BY avec ROLLUP et CUBE sur des relations qui stockent elles-mêmes des valeurs NULL, cela peut produire des jeux de résultats dont les lignes semblent avoir des colonnes de regroupement identiques. Pour revenir à l'exemple précédent, si la colonne de regroupement b contient une valeur NULL stockée, GROUP BY ROLLUP((a), (b)) renvoie une ligne avec une valeur NULL dans la colonne de regroupement b qui n'est pas un sous-total.

Pour faire la distinction entre les valeurs NULL créées par ROLLUP et CUBE et les valeurs NULL stockées dans les tables elles-mêmes, vous pouvez utiliser la fonction GROUPING ou son alias GROUPING_ID. GROUPING prend un seul jeu de regroupement comme argument et, pour chaque ligne du jeu de résultats, renvoie une valeur de 0 ou 1 bit correspondant à la colonne de regroupement dans cette position, puis convertit cette valeur en nombre entier. Si la valeur de cette

position est une valeur NULL créée par une extension d'agrégation, GROUPING renvoie 1. Elle renvoie 0 pour toutes les autres valeurs, y compris les valeurs NULL stockées.

Par exemple, GROUPING(category, product) peut renvoyer les valeurs suivantes pour une ligne donnée, en fonction des valeurs des colonnes de regroupement pour cette ligne. Pour les besoins de cet exemple, toutes les valeurs NULL de la table sont des valeurs NULL créées par une extension d'agrégation.

colonne de catégorie	colonne de produits	Valeur en bits de la fonction GROUPING	Valeur décimale
not NULL	not NULL	00	0
not NULL	NULL	01	1
NULL	not NULL	10	2
NULL	NULL	11	3

Les fonctions GROUPING apparaissent dans la partie liste SELECT de la requête au format suivant.

```
SELECT ... [GROUPING( expr )...] ...
  GROUP BY ... {CUBE | ROLLUP| GROUPING SETS} ( expr ) ...
```

L'exemple suivant est identique à l'exemple précédent pour CUBE, mais avec l'ajout de fonctions GROUPING pour ses jeux de regroupement.

```
SELECT category, product,
       GROUPING(category) as grouping0,
       GROUPING(product) as grouping1,
       GROUPING(category, product) as grouping2,
       sum(cost) as total
FROM orders
GROUP BY CUBE(category, product) ORDER BY 3,1,2;

       category       |       product        | grouping0 | grouping1 | grouping2 |
 total
----------------------+----------------------+-----------+-----------+-----------
+-------
```

```
 cellphones           | smartphone           |         0 |         0 |         0 |
 1610
 cellphones           |                      |         0 |         1 |         1 |
 1610
 computers            | laptop               |         0 |         0 |         0 |
 2050
 computers            | mouse                |         0 |         0 |         0 |
 50
 computers            |                      |         0 |         1 |         1 |
 2100
                      | laptop               |         1 |         0 |         2 |
 2050
                      | mouse                |         1 |         0 |         2 |
 50
                      | smartphone           |         1 |         0 |         2 |
 1610
                      |                      |         1 |         1 |         3 |
 3710
(9 rows)
```

ROLLUP et CUBE partiels

Vous pouvez exécuter les opérations ROLLUP et CUBE avec une partie seulement des sous-totaux.

La syntaxe des opérations ROLLUP et CUBE partielles est la suivante.

```
GROUP BY expr1, { ROLLUP | CUBE }(expr2, [, ...])
```

Ici, la clause GROUP BY crée uniquement des lignes de sous-total au niveau de expr2 et au-delà.

Les exemples suivants illustrent des opérations ROLLUP et CUBE partielles sur la table des commandes, en les regroupant d'abord par produit d'occasion ou non, puis en exécutant ROLLUP et CUBE dans les colonnes de catégorie et de produit.

```
SELECT pre_owned, category, product,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY pre_owned, ROLLUP(category, product) ORDER BY 4,1,2,3;

 pre_owned |       category       |       product        | group_id | total
-----------+----------------------+----------------------+----------+-------
 F         | computers            | laptop               |        0 |  1050
```

```
 F         | computers           | mouse               |        0 |    50
 T         | cellphones          | smartphone          |        0 |  1610
 T         | computers           | laptop              |        0 |  1000
 F         | computers           |                     |        2 |  1100
 T         | cellphones          |                     |        2 |  1610
 T         | computers           |                     |        2 |  1000
 F         |                     |                     |        6 |  1100
 T         |                     |                     |        6 |  2610
(9 rows)

SELECT pre_owned, category, product,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY pre_owned, CUBE(category, product) ORDER BY 4,1,2,3;

 pre_owned |       category       |       product       | group_id | total
-----------+----------------------+---------------------+----------+-------
 F         | computers            | laptop              |        0 |  1050
 F         | computers            | mouse               |        0 |    50
 T         | cellphones           | smartphone          |        0 |  1610
 T         | computers            | laptop              |        0 |  1000
 F         | computers            |                     |        2 |  1100
 T         | cellphones           |                     |        2 |  1610
 T         | computers            |                     |        2 |  1000
 F         |                      | laptop              |        4 |  1050
 F         |                      | mouse               |        4 |    50
 T         |                      | laptop              |        4 |  1000
 T         |                      | smartphone          |        4 |  1610
 F         |                      |                     |        6 |  1100
 T         |                      |                     |        6 |  2610
(13 rows)
```

Comme la colonne d’occasion n’est pas incluse dans les opérations ROLLUP et CUBE, aucune ligne du total général n’inclut toutes les autres lignes.

Concatenated grouping

Vous pouvez concaténer plusieurs SETS/ROLLUP/CUBE clauses GROUPING pour calculer différents niveaux de sous-totaux. Les regroupements concaténés renvoient le produit cartésien des jeux de regroupement fournis.

La syntaxe pour concaténer les SETS/ROLLUP/CUBE clauses GROUPING est la suivante.

```
GROUP BY {ROLLUP|CUBE|GROUPING SETS}(expr1[, ...]),
         {ROLLUP|CUBE|GROUPING SETS}(expr1[, ...])[, ...]
```

Examinez l'exemple suivant pour voir comment un petit regroupement concaténé peut produire un jeu de résultats final volumineux.

```
SELECT pre_owned, category, product,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY CUBE(category, product), GROUPING SETS(pre_owned, ())
ORDER BY 4,1,2,3;

 pre_owned |       category       |       product        | group_id | total
-----------+----------------------+----------------------+----------+-------
 F         | computers            | laptop               |        0 |  1050
 F         | computers            | mouse                |        0 |    50
 T         | cellphones           | smartphone           |        0 |  1610
 T         | computers            | laptop               |        0 |  1000
           | cellphones           | smartphone           |        1 |  1610
           | computers            | laptop               |        1 |  2050
           | computers            | mouse                |        1 |    50
 F         | computers            |                      |        2 |  1100
 T         | cellphones           |                      |        2 |  1610
 T         | computers            |                      |        2 |  1000
           | cellphones           |                      |        3 |  1610
           | computers            |                      |        3 |  2100
 F         |                      | laptop               |        4 |  1050
 F         |                      | mouse                |        4 |    50
 T         |                      | laptop               |        4 |  1000
 T         |                      | smartphone           |        4 |  1610
           |                      | laptop               |        5 |  2050
           |                      | mouse                |        5 |    50
           |                      | smartphone           |        5 |  1610
 F         |                      |                      |        6 |  1100
 T         |                      |                      |        6 |  2610
           |                      |                      |        7 |  3710
(22 rows)
```

Nested grouping

Vous pouvez utiliser les SETS/ROLLUP/CUBE opérations de regroupement en tant qu'expr de vos ENSEMBLES DE GROUPEMENT pour former un regroupement imbriqué. Le sous-regroupement au sein des GROUPING SETS imbriqués est aplati.

La syntaxe pour les regroupements imbriqués est la suivante.

```
GROUP BY GROUPING SETS({ROLLUP|CUBE|GROUPING SETS}(expr[, ...])[, ...])
```

Prenez l'exemple de code suivant.

```
SELECT category, product, pre_owned,
       GROUPING(category, product, pre_owned) as group_id,
       sum(cost) as total
FROM orders
GROUP BY GROUPING SETS(ROLLUP(category), CUBE(product, pre_owned))
ORDER BY 4,1,2,3;

       category       |       product        | pre_owned | group_id | total
----------------------+----------------------+-----------+----------+-------
 cellphones           |                      |           |        3 |  1610
 computers            |                      |           |        3 |  2100
                      | laptop               | F         |        4 |  1050
                      | laptop               | T         |        4 |  1000
                      | mouse                | F         |        4 |    50
                      | smartphone           | T         |        4 |  1610
                      | laptop               |           |        5 |  2050
                      | mouse                |           |        5 |    50
                      | smartphone           |           |        5 |  1610
                      |                      | F         |        6 |  1100
                      |                      | T         |        6 |  2610
                      |                      |           |        7 |  3710
                      |                      |           |        7 |  3710
(13 rows)
```

Notez que, comme ROLLUP(category) et CUBE(product, pre_owned) contiennent tous deux le jeu de regroupement (), la ligne représentant le total général est dupliquée.

Notes d'utilisation

- La clause GROUP BY prend en charge jusqu'à 64 jeux de regroupement. Dans le cas de ROLLUP et CUBE, ou d'une combinaison de GROUPING SETS, ROLLUP et CUBE, cette limitation s'applique au nombre implicite de jeux de regroupement. Par exemple, GROUP BY CUBE((a), (b)) compte comme 4 jeux de regroupement, et non 2.
- Vous ne pouvez pas utiliser de constantes pour regrouper des colonnes lorsque vous utilisez des extensions d'agrégation.
- Vous ne pouvez pas créer un ensemble de regroupement qui contient des colonnes en double.

Clause HAVING

La clause HAVING applique une condition à l'ensemble des résultats groupés intermédiaires que renvoie une requête.

Syntaxe

```
[ HAVING condition ]
```

Par exemple, vous pouvez limiter les résultats d'une fonction SUM :

```
having sum(pricepaid) >10000
```

La condition HAVING est appliquée après que toutes les conditions de la clause WHERE ont été appliquées et que les opérations GROUP BY sont terminées.

La condition elle-même prend la même forme que celle de toute condition de clause WHERE.

Notes d'utilisation

- Toutes les colonnes référencées dans une condition de clause HAVING doivent être une colonne de regroupement ou une colonne qui fait référence au résultat d'une fonction d'agrégation.
- Dans une clause HAVING, vous ne pouvez pas spécifier :
  - Un nombre ordinal qui fait référence à un élément de la liste de sélection. Seules les clauses GROUP BY et ORDER BY acceptent des nombres ordinaux.

Exemples

La requête suivante calcule la vente totale de billets pour tous les événements selon leur nom, puis supprime les événements où le total des ventes est inférieur à 800 000 $ US. La condition HAVING est appliquée aux résultats de la fonction d'agrégation de la liste de sélection : `sum(pricepaid)`.

```
select eventname, sum(pricepaid)
from sales join event on sales.eventid = event.eventid
group by 1
having sum(pricepaid) > 800000
order by 2 desc, 1;

eventname        |    sum
-----------------+-----------
Mamma Mia!       | 1135454.00
Spring Awakening |  972855.00
The Country Girl |  910563.00
Macbeth          |  862580.00
Jersey Boys      |  811877.00
Legally Blonde   |  804583.00
```

La requête suivante calcule un ensemble de résultats similaire. Dans ce cas, toutefois, la condition HAVING est appliquée à un regroupement qui n'est pas spécifié dans la liste de sélection : `sum(qtysold)`. Les événements qui n'ont pas vendu plus de 2 000 billets disparaissent du résultat final.

```
select eventname, sum(pricepaid)
from sales join event on sales.eventid = event.eventid
group by 1
having sum(qtysold) >2000
order by 2 desc, 1;

eventname        |    sum
-----------------+-----------
Mamma Mia!       | 1135454.00
Spring Awakening |  972855.00
The Country Girl |  910563.00
Macbeth          |  862580.00
Jersey Boys      |  811877.00
Legally Blonde   |  804583.00
Chicago          |  790993.00
Spamalot         |  714307.00
```

La requête suivante calcule la vente totale de billets pour tous les événements selon leur nom, puis supprime les événements où le total des ventes est inférieur à 800 000 $ US. La condition HAVING est appliquée aux résultats de la fonction d'agrégation dans la liste de sélection à l'aide de l'alias pp for`sum(pricepaid)`.

```
select eventname, sum(pricepaid) as pp
from sales join event on sales.eventid = event.eventid
group by 1
having pp > 800000
order by 2 desc, 1;

eventname        |    pp
-----------------+-----------
Mamma Mia!       | 1135454.00
Spring Awakening |  972855.00
The Country Girl |  910563.00
Macbeth          |  862580.00
Jersey Boys      |  811877.00
Legally Blonde   |  804583.00
```

Clause QUALIFY

La clause QUALIFY filtre les résultats d'une fonction de fenêtre précédemment calculée en fonction des conditions de recherche définies par l'utilisateur. Vous pouvez utiliser la clause pour appliquer des conditions de filtrage au résultat d'une fonction de fenêtre sans utiliser de sous-requête.

Elle est similaire à la clause HAVING, qui applique une condition pour filtrer davantage les lignes d'une clause WHERE. La différence entre QUALIFY et HAVING réside dans le fait que les résultats filtrés de la clause QUALIFY peuvent être basés sur le résultat de l'exécution de fonctions de fenêtre sur les données. Vous pouvez utiliser à la fois les clauses QUALIFY et HAVING dans une même requête.

Syntaxe

```
QUALIFY condition
```

> **Note**
>
> Si vous utilisez la clause QUALIFY directement après la clause FROM, le nom de la relation FROM doit comporter un alias spécifié avant la clause QUALIFY.

Exemples

Les exemples de cette section utilisent les exemples de données ci-dessous.

```
create table store_sales (ss_sold_date date, ss_sold_time time,
               ss_item text, ss_sales_price float);
insert into store_sales values ('2022-01-01', '09:00:00', 'Product 1', 100.0),
                               ('2022-01-01', '11:00:00', 'Product 2', 500.0),
                               ('2022-01-01', '15:00:00', 'Product 3', 20.0),
                               ('2022-01-01', '17:00:00', 'Product 4', 1000.0),
                               ('2022-01-01', '18:00:00', 'Product 5', 30.0),
                               ('2022-01-02', '10:00:00', 'Product 6', 5000.0),
                               ('2022-01-02', '16:00:00', 'Product 7', 5.0);
```

L'exemple suivant montre comment trouver les deux articles les plus chers vendus après midi chaque jour.

```
SELECT *
FROM store_sales ss
WHERE ss_sold_time > time '12:00:00'
QUALIFY row_number()
OVER (PARTITION BY ss_sold_date ORDER BY ss_sales_price DESC) <= 2


 ss_sold_date | ss_sold_time |  ss_item  | ss_sales_price
--------------+--------------+-----------+----------------
 2022-01-01   | 17:00:00     | Product 4 |           1000
 2022-01-01   | 18:00:00     | Product 5 |             30
 2022-01-02   | 16:00:00     | Product 7 |              5
```

Vous pourrez alors retrouver le dernier article vendu chaque jour.

```
SELECT *
FROM store_sales ss
QUALIFY last_value(ss_item)
OVER (PARTITION BY ss_sold_date ORDER BY ss_sold_time ASC
      ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) = ss_item;

ss_sold_date | ss_sold_time |  ss_item  | ss_sales_price
--------------+--------------+-----------+----------------
 2022-01-01   | 18:00:00     | Product 5 |             30
 2022-01-02   | 16:00:00     | Product 7 |              5
```

L'exemple suivant renvoie les mêmes enregistrements que la requête précédente, le dernier article vendu chaque jour, mais il n'utilise pas la clause QUALIFY.

```
SELECT * FROM (
  SELECT *,
  last_value(ss_item)
  OVER (PARTITION BY ss_sold_date ORDER BY ss_sold_time ASC
        ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) ss_last_item
  FROM store_sales ss
)
WHERE ss_last_item = ss_item;

 ss_sold_date | ss_sold_time |  ss_item  | ss_sales_price | ss_last_item
--------------+--------------+-----------+----------------+--------------
 2022-01-02   | 16:00:00     | Product 7 |              5 | Product 7
 2022-01-01   | 18:00:00     | Product 5 |             30 | Product 5
```

UNION, INTERSECT et EXCEPT

Rubriques

- Syntaxe
- Paramètres
- Ordre d’évaluation des opérateurs ensemblistes
- Notes d’utilisation
- Exemple de requêtes UNION
- Exemple de requête UNION ALL
- Exemple de requêtes INTERSECT
- Exemple de requête EXCEPT

Les opérateurs ensemblistes UNION, INTERSECT et EXCEPT sont utilisés pour comparer et fusionner les résultats de deux expressions de requête distinctes. Par exemple, si vous voulez savoir quels utilisateurs d’un site web sont à la fois acheteurs et vendeurs, mais que leurs noms d’utilisateur sont stockés dans des colonnes ou tables distinctes, vous pouvez trouver l’intersection de ces deux types d’utilisateurs. Si vous voulez savoir quels utilisateurs du site web sont acheteurs mais pas vendeurs, vous pouvez utiliser l’opérateur EXCEPT pour trouver la différence entre les deux listes d’utilisateurs. Si vous souhaitez créer une liste de tous les utilisateurs, quel que soit le rôle, vous pouvez utiliser l’opérateur UNION.

Syntaxe

```
query
{ UNION [ ALL ] | INTERSECT | EXCEPT | MINUS }
query
```

Paramètres

query

Expression de requête qui correspond, sous la forme de sa liste de sélection, à une deuxième expression de requête qui suit l'opérateur UNION, INTERSECT ou EXCEPT. Les deux expressions doivent comporter le même nombre de colonnes de sortie avec des types de données compatibles ; sinon, les deux jeux de résultats ne peuvent pas être comparés et fusionnés. Les opérations définies n'autorisent pas la conversion implicite entre différentes catégories de types de données ; pour plus d'informations, consultez Compatibilité et conversion de types.

Vous pouvez créer des requêtes qui contiennent un nombre illimité d'expressions de requête et les lier avec les opérateurs UNION, INTERSECT et EXCEPT dans n'importe quelle combinaison. Par exemple, la structure de requête suivante est valide, en supposant que les tables T1, T2 et T3 contiennent des ensembles de colonnes compatibles :

```
select * from t1
union
select * from t2
except
select * from t3
order by c1;
```

UNION

Opération de définition qui renvoie les lignes de deux expressions de requête, indépendamment de savoir si les lignes proviennent de l'une ou des deux expressions.

INTERSECT

Opération de définition qui renvoie les lignes provenant de deux expressions de requête. Les lignes qui ne sont pas retournées par les deux expressions sont ignorées.

EXCEPT | MINUS

Opération de définition qui renvoie les lignes qui dérivent de l'une de deux expressions de requête. Pour être éligible pour le résultat, lignes doivent exister dans la première table de résultats, pas dans la deuxième. MINUS et EXCEPT sont des synonymes exacts.

ALL

Le mot-clé ALL conserve toutes les lignes en double produites par UNION. Le comportement par défaut lorsque le mot-clé ALL n'est pas utilisé consiste à ignorer ces doublons. INTERSECT ALL, EXCEPT ALL et MINUS ALL ne sont pas pris en charge.

Ordre d'évaluation des opérateurs ensemblistes

Les opérateurs ensemblistes UNION et EXCEPT sont associatifs à gauche. Si les parenthèses ne sont pas spécifiées pour influer sur l'ordre de priorité, une combinaison de ces opérateurs ensemblistes est évaluée de gauche à droite. Par exemple, dans la requête suivante, l'UNION de T1 et de T2 est évaluée en premier, puis l'opération EXCEPT est effectuée sur le résultat UNION :

```
select * from t1
union
select * from t2
except
select * from t3
order by c1;
```

L'opérateur INTERSECT est prioritaire sur les opérateurs UNION et EXCEPT quand une combinaison d'opérateurs est utilisée dans la même requête. Par exemple, la requête suivante permet d'évaluer l'intersection de T2 et de T3, puis d'unir le résultat à T1 :

```
select * from t1
union
select * from t2
intersect
select * from t3
order by c1;
```

Par l'ajout de parenthèses, vous pouvez appliquer un ordre d'évaluation différent. Dans le cas suivant, le résultat de l'union de T1 et de T2 est croisé avec T3, et la requête est susceptible de produire un résultat différent.

```
(select * from t1
union
select * from t2)
intersect
(select * from t3)
order by c1;
```

Notes d'utilisation

- Les noms de colonne retournés dans le résultat d'une opération ensembliste sont les noms de colonne (ou alias) des tables de la première expression de requête. Comme ces noms de colonne sont potentiellement trompeurs, en ce sens que les valeurs de la colonne proviennent de tables de l'un ou de l'autre côté de l'opérateur ensembliste, il se peut que vous vouliez fournir des alias descriptifs pour le jeu de résultats.
- Une expression de requête qui précède un opérateur ensembliste ne doit pas contenir une clause ORDER BY. Une clause ORDER BY ne produit des résultats triés significatifs que lorsqu'elle est utilisée à la fin d'une requête contenant des opérateurs ensemblistes. Dans ce cas, la clause ORDER BY s'applique aux résultats finaux de toutes les opérations ensemblistes. La requête la plus externe peut également contenir des clauses LIMIT et OFFSET standard.
- Lorsque les requêtes avec opérateurs ensemblistes renvoient des résultats décimaux, les colonnes de résultats correspondantes sont promues pour renvoyer les mêmes précision et échelle. Par exemple, dans la requête suivante, où T1.REVENUE est une colonne DECIMAL(10,2) et T2.REVENUE une colonne DECIMAL(8,4), le résultat décimal est promu en DECIMAL(12,4) :

  ```
  select t1.revenue union select t2.revenue;
  ```

  L'échelle est 4, parce que c'est l'échelle maximale des deux colonnes. La précision est 12 parce que T1.REVENUE nécessite 8 chiffres à gauche de la virgule (12-4 = 8). Cette promotion de type garantit que toutes les valeurs de chaque côté de l'UNION conviennent au résultat. Pour les valeurs 64 bits, la précision de résultat maximale est de 19 et l'échelle de résultat maximale de 18. Pour les valeurs 128 bits, la précision de résultat maximale est de 38 et l'échelle de résultat maximale de 37.

  Si le type de données qui en résulte dépasse les limites Amazon Redshift de précision et d'échelle, la requête renvoie une erreur.
- Pour les opérations ensemblistes, deux lignes sont traitées comme identiques si, pour chaque paire correspondante de colonnes, les deux valeurs de données sont égales ou toutes deux NULL.

Par exemple, si les tables T1 et T2 contiennent une colonne et une ligne, et que la ligne a la valeur NULL dans les deux tables, une opération INTERSECT sur ces tables renvoie cette ligne.

Exemple de requêtes UNION

Dans la requête UNION suivante, les lignes de la table SALES sont fusionnées avec les lignes de la table LISTING. Trois colonnes compatibles sont sélectionnées à partir de chaque table ; dans ce cas, les colonnes correspondantes ont les mêmes noms et types de données.

L'ensemble de résultats final est classé sur la première colonne de la table LISTING et limité aux 5 lignes avec la valeur LISTID la plus élevée.

```
select listid, sellerid, eventid from listing
union select listid, sellerid, eventid from sales
order by listid, sellerid, eventid desc limit 5;

listid | sellerid | eventid
--------+----------+---------
1 |    36861 |    7872
2 |    16002 |    4806
3 |    21461 |    4256
4 |     8117 |    4337
5 |     1616 |    8647
(5 rows)
```

L'exemple suivant montre comment vous pouvez ajouter une valeur littérale à la sortie d'une requête UNION afin que vous puissiez voir quelle expression de requête a généré chaque ligne du jeu de résultats. La requête identifie les lignes de la première expression de requête comme « B » (pour « buyers ») et les lignes de la deuxième expression de requête comme « S » (pour « sellers »).

La requête identifie les acheteurs et les vendeurs pour les transactions de billet égales ou supérieures à 10 000 $ US. La seule différence entre les deux expressions de requête de chaque côté de l'opérateur d'UNION est la colonne de jointure de la table SALES.

```
select listid, lastname, firstname, username,
pricepaid as price, 'S' as buyorsell
from sales, users
where sales.sellerid=users.userid
and pricepaid >=10000
union
select listid, lastname, firstname, username, pricepaid,
```

```
'B' as buyorsell
from sales, users
where sales.buyerid=users.userid
and pricepaid >=10000
order by 1, 2, 3, 4, 5;

listid | lastname | firstname | username |   price   | buyorsell
--------+----------+-----------+----------+-----------+-----------
209658 | Lamb     | Colette   | VOR15LYI |  10000.00 | B
209658 | West     | Kato      | ELU81XAA |  10000.00 | S
212395 | Greer    | Harlan    | GXO71KOC |  12624.00 | S
212395 | Perry    | Cora      | YWR73YNZ |  12624.00 | B
215156 | Banks    | Patrick   | ZNQ69CLT |  10000.00 | S
215156 | Hayden   | Malachi   | BBG56AKU |  10000.00 | B
(6 rows)
```

L’exemple suivant utilise un opérateur UNION ALL, car les lignes dupliquées, s’il en existe, doivent être conservées dans le résultat. Pour une série d'événements spécifique IDs, la requête renvoie 0 ligne ou plus pour chaque vente associée à chaque événement, et 0 ou 1 ligne pour chaque annonce de cet événement. IDs Les événements sont propres à chaque ligne des tableaux LISTING et EVENT, mais il peut y avoir plusieurs ventes pour la même combinaison d'événement et d'annonce IDs dans le tableau SALES.

La troisième colonne du jeu de résultats identifie la source de la ligne. Si la source est la table SALES, un « Yes » apparaît dans la colonne SALESROW. (SALESROW est un alias de SALES. LISTID.) Si la ligne vient de la table LISTING, un « No » apparaît dans la colonne SALESROW.

Dans ce cas, le jeu de résultats se compose de trois lignes de vente pour affichage 500, événement 7787. En d’autres termes, trois transactions différentes ont eu lieu pour cette combinaison d’affichage et d’événement. Les deux autres listes, 501 et 502, n'ont généré aucune vente. La seule ligne produite par la requête pour ces listes IDs provient donc de la table LISTING (SALESROW = « Non »).

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union all
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;
```

```
eventid | listid | salesrow
---------+--------+----------
7787 |    500 | No
7787 |    500 | Yes
7787 |    500 | Yes
7787 |    500 | Yes
6473 |    501 | No
5108 |    502 | No
(6 rows)
```

Si vous exécutez la même requête sans le mot-clé ALL, le résultat ne conserve qu'une seule des transactions de vente.

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
7787 |    500 | No
7787 |    500 | Yes
6473 |    501 | No
5108 |    502 | No
(4 rows)
```

Exemple de requête UNION ALL

L'exemple suivant utilise un opérateur UNION ALL, car les lignes dupliquées, s'il en existe, doivent être conservées dans le résultat. Pour une série d'événements spécifique IDs, la requête renvoie 0 ligne ou plus pour chaque vente associée à chaque événement, et 0 ou 1 ligne pour chaque annonce de cet événement. IDs Les événements sont propres à chaque ligne des tableaux LISTING et EVENT, mais il peut y avoir plusieurs ventes pour la même combinaison d'événement et d'annonce IDs dans le tableau SALES.

La troisième colonne du jeu de résultats identifie la source de la ligne. Si la source est la table SALES, un « Yes » apparaît dans la colonne SALESROW. (SALESROW est un alias de SALES. LISTID.) Si la ligne vient de la table LISTING, un « No » apparaît dans la colonne SALESROW.

Dans ce cas, le jeu de résultats se compose de trois lignes de vente pour affichage 500, événement 7787. En d'autres termes, trois transactions différentes ont eu lieu pour cette combinaison d'affichage et d'événement. Les deux autres listes, 501 et 502, n'ont généré aucune vente. La seule ligne produite par la requête pour ces listes IDs provient donc de la table LISTING (SALESROW = « Non »).

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union all
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
7787 |    500 | No
7787 |    500 | Yes
7787 |    500 | Yes
7787 |    500 | Yes
6473 |    501 | No
5108 |    502 | No
(6 rows)
```

Si vous exécutez la même requête sans le mot-clé ALL, le résultat ne conserve qu'une seule des transactions de vente.

```
select eventid, listid, 'Yes' as salesrow
from sales
where listid in(500,501,502)
union
select eventid, listid, 'No'
from listing
where listid in(500,501,502)
order by listid asc;

eventid | listid | salesrow
---------+--------+----------
7787 |    500 | No
7787 |    500 | Yes
6473 |    501 | No
```

```
5108 |    502 | No
(4 rows)
```

Exemple de requêtes INTERSECT

Comparez l'exemple suivant avec le premier exemple UNION. La seule différence entre les deux exemples est l'opérateur ensembliste qui est utilisé, mais les résultats sont très différents. Seule une des lignes est la même :

```
235494 |    23875 |    8771
```

Il s'agit de la seule ligne du résultat limité de 5 lignes qui a été trouvé dans les deux tables.

```
select listid, sellerid, eventid from listing
intersect
select listid, sellerid, eventid from sales
order by listid desc, sellerid, eventid
limit 5;

listid | sellerid | eventid
--------+----------+---------
235494 |    23875 |    8771
235482 |     1067 |    2667
235479 |     1589 |    7303
235476 |    15550 |     793
235475 |    22306 |    7848
(5 rows)
```

La requête suivante détecte les événements (pour lesquels des billets ont été vendus) qui se sont déroulées dans des lieux de New York et de Los Angeles en mars. La différence entre les deux expressions de requête est la contrainte sur la colonne VENUECITY.

```
select distinct eventname from event, sales, venue
where event.eventid=sales.eventid and event.venueid=venue.venueid
and date_part(month,starttime)=3 and venuecity='Los Angeles'
intersect
select distinct eventname from event, sales, venue
where event.eventid=sales.eventid and event.venueid=venue.venueid
and date_part(month,starttime)=3 and venuecity='New York City'
order by eventname asc;

eventname
```

```
----------------------------
A Streetcar Named Desire
Dirty Dancing
Electra
Running with Annalise
Hairspray
Mary Poppins
November
Oliver!
Return To Forever
Rhinoceros
South Pacific
The 39 Steps
The Bacchae
The Caucasian Chalk Circle
The Country Girl
Wicked
Woyzeck
(16 rows)
```

Exemple de requête EXCEPT

La table CATEGORY de la base de données TICKIT contient les 11 lignes suivantes :

```
 catid | catgroup |  catname  |                 catdesc
-------+----------+-----------+--------------------------------------------
   1   | Sports   | MLB       | Major League Baseball
   2   | Sports   | NHL       | National Hockey League
   3   | Sports   | NFL       | National Football League
   4   | Sports   | NBA       | National Basketball Association
   5   | Sports   | MLS       | Major League Soccer
   6   | Shows    | Musicals  | Musical theatre
   7   | Shows    | Plays     | All non-musical theatre
   8   | Shows    | Opera     | All opera and light opera
   9   | Concerts | Pop       | All rock and pop music concerts
  10   | Concerts | Jazz      | All jazz singers and bands
  11   | Concerts | Classical | All symphony, concerto, and choir concerts
(11 rows)
```

Supposons qu'une table CATEGORY_STAGE (table intermédiaire) contienne une seule ligne supplémentaire :

```
 catid | catgroup |  catname  |                 catdesc
```

```
-------+----------+-----------+--------------------------------------------
1 | Sports   | MLB       | Major League Baseball
2 | Sports   | NHL       | National Hockey League
3 | Sports   | NFL       | National Football League
4 | Sports   | NBA       | National Basketball Association
5 | Sports   | MLS       | Major League Soccer
6 | Shows    | Musicals  | Musical theatre
7 | Shows    | Plays     | All non-musical theatre
8 | Shows    | Opera     | All opera and light opera
9 | Concerts | Pop       | All rock and pop music concerts
10 | Concerts | Jazz      | All jazz singers and bands
11 | Concerts | Classical | All symphony, concerto, and choir concerts
12 | Concerts | Comedy    | All stand up comedy performances
(12 rows)
```

renvoiez la différence entre les deux tables. En d'autres termes, renvoiez les lignes qui sont dans la table CATEGORY_STAGE, mais pas dans la table CATEGORY :

```
select * from category_stage
except
select * from category;

catid | catgroup | catname |             catdesc
-------+----------+---------+----------------------------------
12 | Concerts | Comedy  | All stand up comedy performances
(1 row)
```

La requête équivalente suivante utilise le synonyme MINUS.

```
select * from category_stage
minus
select * from category;

catid | catgroup | catname |             catdesc
-------+----------+---------+----------------------------------
12 | Concerts | Comedy  | All stand up comedy performances
(1 row)
```

Si vous inversez l'ordre des expressions SELECT, la requête ne renvoie aucune ligne.

Clause ORDER BY

Rubriques

- Syntaxe
- Paramètres
- Notes d'utilisation
- Exemples avec ORDER BY

La clause ORDER BY trie le jeu de résultats d'une requête.

Syntaxe

```
[ ORDER BY expression [ ASC | DESC ] ]
[ NULLS FIRST | NULLS LAST ]
[ LIMIT { count | ALL } ]
[ OFFSET start ]
```

Paramètres

expression

Expression qui définit l'ordre de tri du jeu de résultats de la requête, généralement en spécifiant une ou plusieurs colonnes de la liste de sélection. Les résultats sont retournés en fonction du classement UTF-8 binaire. Vous pouvez aussi spécifier les éléments suivants :

- Colonnes qui ne sont pas dans la liste de sélection
- Expressions formées d'une ou de plusieurs colonnes qui existent dans les tables référencées par la requête
- Nombres ordinaux qui représentent la position des entrées de la liste de sélection (ou position des colonnes de la table s'il n'existe aucune liste de sélection)
- Alias qui définissent les entrées de la liste de sélection

Lorsque la clause ORDER BY contient plusieurs expressions régulières, le jeu de résultats est trié selon la première expression, puis la deuxième expression est appliquée aux lignes de la première expression ayant des valeurs correspondantes, et ainsi de suite.

ASC | DESC

Option qui définit l'ordre de tri de l'expression, comme suit :

- ASC : croissant (par exemple, de faible à élevé pour les valeurs numériques et de « A » à « Z » pour les chaînes de caractères). Si aucune option n'est spécifiée, les données sont triées dans l'ordre croissant par défaut.
- DESC : descendantes (valeurs d'élevées à faibles pour les valeurs numériques ; de « Z » à « A » pour les chaînes).

NULLS FIRST | NULLS LAST

Option qui spécifie si les valeurs NULL doivent être triées en premier, avant les valeurs non null, ou en dernier, après les valeurs non null. Par défaut, les valeurs NULL sont triées et classées en dernier par ordre croissant (ASC) et triées et classées en premier par ordre décroissant (DESC).

LIMIT nombre | ALL

Option qui contrôle le nombre de lignes triées renvoyées par la requête. Le nombre LIMIT doit être un nombre entier positif ; la valeur maximale est 2147483647.

LIMIT 0 ne renvoie aucune ligne. Vous pouvez utiliser cette syntaxe à des fins de test, pour vérifier qu'une requête s'exécute (sans afficher aucune ligne) ou pour renvoyer une liste de colonnes d'une table. Une clause ORDER BY est redondante si vous utilisez LIMIT 0 pour renvoyer une liste de colonnes. La valeur par défaut est LIMIT ALL.

OFFSET début

Option qui spécifie d'ignorer le nombre de lignes qui précèdent début avant de commencer à renvoyer les lignes. Le nombre OFFSET doit être un nombre entier positif ; la valeur maximale est 2147483647. Lorsqu'elles sont utilisées avec l'option LIMIT, les lignes OFFSET sont ignorées avant de commencer à compter les lignes LIMIT qui sont retournées. Si l'option LIMIT n'est pas utilisée, le nombre de lignes du jeu de résultats est diminué du nombre de lignes qui sont ignorées. Comme les lignes ignorées par une clause OFFSET continuent de devoir être analysées, il peut être inefficace de choisir une valeur OFFSET élevée.

Notes d'utilisation

Notez le comportement attendu suivant avec les clauses ORDER BY :

- Les valeurs NULL sont considérées comme « plus élevés » que toutes les autres valeurs. Avec l'ordre de tri croissant par défaut, les valeurs NULL sont triées à la fin. Pour modifier ce comportement, utilisez l'option NULLS FIRST.
- Lorsqu'une requête ne contient pas une clause ORDER BY, le système renvoie des jeux de résultats sans classement prévisible des lignes. La même requête exécutée deux fois peut renvoyer le même jeu de résultats dans un ordre différent.
- Les options LIMIT et OFFSET peuvent être utilisées sans clause ORDER BY ; cependant, pour renvoyer un ensemble cohérent de lignes, utilisez ces options conjointement à ORDER BY.
- Dans tout système parallèle comme Amazon Redshift, quand ORDER BY ne produit pas un classement unique, l'ordre des lignes est non déterministe. Autrement dit, si l'expression ORDER BY produit des valeurs en double, l'ordre de retour de ces lignes peut varier d'un système à un autre ou d'une exécution d'Amazon Redshift à une autre.
- Amazon Redshift ne prend pas en charge les littéraux de chaîne dans les clauses ORDER BY.

Exemples avec ORDER BY

renvoiez les 11 lignes de la table CATEGORY, triées sur la deuxième colonne, CATGROUP. Pour les résultats qui ont la même valeur CATGROUP, classez les valeurs de colonne CATDESC en fonction de la longueur de la chaîne de caractères. Triez ensuite sur les colonnes CATID et CATNAME.

```
select * from category order by 2, length(catdesc), 1, 3;

catid | catgroup |  catname  |                 catdesc
------+----------+-----------+--------------------------------------
10    | Concerts | Jazz      | All jazz singers and bands
9     | Concerts | Pop       | All rock and pop music concerts
11    | Concerts | Classical | All symphony, concerto, and choir conce
6     | Shows    | Musicals  | Musical theatre
7     | Shows    | Plays     | All non-musical theatre
8     | Shows    | Opera     | All opera and light opera
5     | Sports   | MLS       | Major League Soccer
1     | Sports   | MLB       | Major League Baseball
2     | Sports   | NHL       | National Hockey League
3     | Sports   | NFL       | National Football League
4     | Sports   | NBA       | National Basketball Association
(11 rows)
```

renvoiez les colonnes sélectionnées de la table SALES, triées selon les valeurs QTYSOLD les plus élevées. Limitez les résultats aux 10 lignes supérieures :

```
select salesid, qtysold, pricepaid, commission, saletime from sales
order by qtysold, pricepaid, commission, salesid, saletime desc
limit 10;

salesid | qtysold | pricepaid | commission |      saletime
--------+---------+-----------+------------+---------------------
15401   |       8 |    272.00 |      40.80 | 2008-03-18 06:54:56
61683   |       8 |    296.00 |      44.40 | 2008-11-26 04:00:23
90528   |       8 |    328.00 |      49.20 | 2008-06-11 02:38:09
74549   |       8 |    336.00 |      50.40 | 2008-01-19 12:01:21
130232  |       8 |    352.00 |      52.80 | 2008-05-02 05:52:31
55243   |       8 |    384.00 |      57.60 | 2008-07-12 02:19:53
16004   |       8 |    440.00 |      66.00 | 2008-11-04 07:22:31
489     |       8 |    496.00 |      74.40 | 2008-08-03 05:48:55
4197    |       8 |    512.00 |      76.80 | 2008-03-23 11:35:33
16929   |       8 |    568.00 |      85.20 | 2008-12-19 02:59:33
(10 rows)
```

renvoiez une liste de colonnes et aucune ligne à l'aide de la syntaxe LIMIT 0 :

```
select * from venue limit 0;
venueid | venuename | venuecity | venuestate | venueseats
---------+-----------+-----------+------------+------------
(0 rows)
```

Clause CONNECT BY

La clause CONNECT BY spécifie la relation entre les lignes d'une hiérarchie. Vous pouvez utiliser CONNECT BY pour sélectionner des lignes dans un ordre hiérarchique en attachant la table à elle-même et en traitant les données hiérarchiques. Par exemple, vous pouvez l'utiliser pour parcourir de manière récursive un organigramme et répertorier des données.

Les requêtes hiérarchiques sont traitées dans l'ordre suivant :

1. Si la clause FROM comporte une jointure, elle est traitée en premier.
2. La clause CONNECT BY est évaluée.
3. La clause WHERE est évaluée.

Syntaxe

```
[START WITH start_with_conditions]
CONNECT BY connect_by_conditions
```

> **Note**
>
> Bien que START et CONNECT ne soient pas des mots réservés, utilisez des identificateurs délimités (guillemets doubles) ou AS si vous utilisez START et CONNECT comme alias de table dans votre requête, afin d'éviter tout échec lors de l'exécution.

```
SELECT COUNT(*)
FROM Employee "start"
CONNECT BY PRIOR id = manager_id
START WITH name = 'John'
```

```
SELECT COUNT(*)
FROM Employee AS start
CONNECT BY PRIOR id = manager_id
START WITH name = 'John'
```

Paramètres

start_with_conditions

Conditions qui spécifient la ou les lignes racines de la hiérarchie

connect_by_conditions

Conditions qui spécifient la relation entre les lignes parents et les lignes enfants de la hiérarchie. Au moins une condition doit être qualifiée à l'aide de l'opérateur unaire utilisé pour faire référence à la ligne parent.

```
PRIOR column = expression
-- or
expression > PRIOR column
```

Opérateurs

Vous pouvez utiliser les opérateurs suivants dans une requête CONNECT BY.

LEVEL

Pseudocolonne qui renvoie le niveau de ligne actuel dans la hiérarchie. Renvoie 1 pour la ligne racine, 2 pour l'enfant de la ligne racine, etc.

PRIOR

Opérateur unaire qui évalue l'expression pour la ligne parent de la ligne actuelle dans la hiérarchie.

Exemples

Voici un exemple de requête CONNECT BY qui renvoie le nombre d'employés qui relèvent directement ou indirectement de John, avec 4 niveaux maximum.

```
SELECT id, name, manager_id
FROM employee
WHERE LEVEL < 4
START WITH name = 'John'
CONNECT BY PRIOR id = manager_id;
```

Voici le résultat de la requête.

```
id      name      manager_id
------+----------+--------------
  101     John        100
  102     Jorge       101
  103     Kwaku       101
  110     Liu         101
  201     Sofía       102
  106     Mateo       102
  110     Nikki       103
  104     Paulo       103
  105     Richard     103
  120     Saanvi      104
  200     Shirley     104
  205     Zhang       104
```

Définition de table pour cet exemple :

```
CREATE TABLE employee (
   id INT,
   name VARCHAR(20),
   manager_id INT
   );
```

Voici les lignes insérées dans la table.

```
INSERT INTO employee(id, name, manager_id)  VALUES
(100, 'Carlos', null),
(101, 'John', 100),
(102, 'Jorge', 101),
(103, 'Kwaku', 101),
(110, 'Liu', 101),
(106, 'Mateo', 102),
(110, 'Nikki', 103),
(104, 'Paulo', 103),
(105, 'Richard', 103),
(120, 'Saanvi', 104),
(200, 'Shirley', 104),
(201, 'Sofía', 102),
(205, 'Zhang', 104);
```

Voici un organigramme du service de John.

Exemples de sous-requête

Les exemples suivants illustrent différentes façons par lesquelles les sous-requêtes conviennent aux requêtes SELECT. Pour obtenir un autre exemple de l'utilisation des sous-requêtes, consultez Exemples de clause JOIN.

Sous-requête SELECT liste

L'exemple suivant contient une sous-requête dans la liste SELECT. Cette sous-requête est scalaire : elle renvoie une et une seule colonne et une seule valeur, ce qui est répété dans le résultat pour chaque ligne retournée à partir de la requête externe. La requête compare la valeur Q1SALES que la sous-requête calcule aux valeurs des ventes des deux autres trimestres (2 et 3) en 2008, comme défini par la requête externe.

```
select qtr, sum(pricepaid) as qtrsales,
(select sum(pricepaid)
from sales join date on sales.dateid=date.dateid
where qtr='1' and year=2008) as q1sales
from sales join date on sales.dateid=date.dateid
where qtr in('2','3') and year=2008
group by qtr
order by qtr;

qtr  |  qtrsales   |   q1sales
-------+-------------+-------------
2     | 30560050.00 | 24742065.00
3     | 31170237.00 | 24742065.00
(2 rows)
```

Sous-requête de clause WHERE

L'exemple suivant contient une sous-requête de table dans la clause WHERE. Cette sous-requête produit plusieurs lignes. Dans ce cas, les lignes ne contiennent qu'une seule colonne, mais les sous-requêtes de table peuvent contenir plusieurs colonnes et lignes, tout comme n'importe quelle autre table.

La requête recherche les 10 meilleurs vendeurs en termes de nombre maximal de billets vendus. La liste des 10 meilleurs est limitée par la sous-requête, qui supprime les utilisateurs qui résident dans les villes où il y a des lieux de vente. Cette requête peut être écrite de différentes façons ; par exemple, la sous-requête peut être réécrite comme jointure au sein de la requête principale.

```
select firstname, lastname, city, max(qtysold) as maxsold
from users join sales on users.userid=sales.sellerid
where users.city not in(select venuecity from venue)
group by firstname, lastname, city
order by maxsold desc, city desc
limit 10;

firstname | lastname  |      city      | maxsold
-----------+-----------+----------------+---------
Noah       | Guerrero | Worcester      |       8
Isadora    | Moss     | Winooski       |       8
Kieran     | Harrison | Westminster    |       8
Heidi      | Davis    | Warwick        |       8
Sara       | Anthony  | Waco           |       8
Bree       | Buck     | Valdez         |       8
```

```
Evangeline | Sampson  | Trenton        |       8
Kendall    | Keith    | Stillwater     |       8
Bertha     | Bishop   | Stevens Point  |       8
Patricia   | Anderson | South Portland |       8
(10 rows)
```

Sous-requêtes de clause WITH

Consultez Clause WITH.

Sous-requêtes corrélées

L'exemple suivant contient une sous-requête corrélée dans la clause WHERE ; ce genre de sous-requête contient une ou plusieurs corrélations entre ses colonnes et les colonnes générés par la requête externe. Dans ce cas, la corrélation est `where s.listid=l.listid`. Pour chaque ligne que produit la requête externe, la sous-requête est exécutée pour qualifier ou disqualifier la ligne.

```
select salesid, listid, sum(pricepaid) from sales s
where qtysold=
(select max(numtickets) from listing l
where s.listid=l.listid)
group by 1,2
order by 1,2
limit 5;

salesid | listid |   sum
--------+--------+----------
 27     |     28 | 111.00
 81     |    103 | 181.00
 142    |    149 | 240.00
 146    |    152 | 231.00
 194    |    210 | 144.00
(5 rows)
```

Modèles de sous-requêtes corrélées non pris en charge

Le planificateur de requête utilise une méthode de réécriture de requête appelée décorrélation de sous-requête afin d'optimiser plusieurs modèles de sous-requêtes corrélées en vue de l'exécution dans un environnement MPP. Quelques types de sous-requêtes corrélées suivent des modèles qu'Amazon Redshift ne peut pas décorréler et ne prend pas en charge. Les requêtes qui contiennent les références de corrélation suivantes génèrent des erreurs :

- Les références de corrélation qui ignorent un bloc de requête, également appelées « références de corrélation de niveau non hiérarchique ». Par exemple, dans la requête suivante, le bloc contenant la référence de corrélation et le bloc ignoré sont connectés par un prédicat NOT EXISTS :

```
select event.eventname from event
where not exists
(select * from listing
where not exists
(select * from sales where event.eventid=sales.eventid));
```

  Le bloc ignoré dans ce cas est la sous-requête sur la table LISTING. La référence de corrélation correspond aux tables EVENT et SALES.

- Références de corrélation à partir d'une sous-requête qui fait partie d'une clause ON dans une requête externe :

```
select * from category
left join event
on category.catid=event.catid and eventid =
(select max(eventid) from sales where sales.eventid=event.eventid);
```

  La clause ON contient une référence de corrélation depuis SALES dans la sous-requête jusqu'à EVENT dans la requête externe.

- Références de corrélation sensibles à null à une table système Amazon Redshift. Par exemple :

```
select attrelid
from stv_locks sl, pg_attribute
where sl.table_id=pg_attribute.attrelid and 1 not in
(select 1 from pg_opclass where sl.lock_owner = opcowner);
```

- Références de corrélation à partir d'une sous-requête contenant une fonction de fenêtrage.

```
select listid, qtysold
from sales s
where qtysold not in
(select sum(numtickets) over() from listing l where s.listid=l.listid);
```

- Références d'une colonne GROUP BY aux résultats d'une sous-requête corrélée. Par exemple :

```
select listing.listid,
(select count (sales.listid) from sales where sales.listid=listing.listid) as list
```

```
from listing
group by list, listing.listid;
```

- Références de corrélation à partir d'une sous-requête avec fonction d'agrégation et d'une clause GROUP BY, connectée à la requête externe par un prédicat IN. (Cette restriction ne s'applique pas aux fonctions d'agrégation MIN et MAX.) Par exemple :

```
select * from listing where listid in
(select sum(qtysold)
from sales
where numtickets>4
group by salesid);
```

SELECT INTO

Sélectionne les lignes définies par une requête et les insère dans une nouvelle table. Vous pouvez spécifier s'il faut créer une table temporaire ou permanente.

Syntaxe

```
[ WITH with_subquery [, ...] ]
SELECT
[ TOP number | [ ALL | DISTINCT ]
* | expression [ AS output_name ] [, ...] ]
[ EXCLUDE column_list ]
INTO [ TEMPORARY | TEMP ] [ TABLE ] new_table
[ FROM table_reference [, ...] ]
[ WHERE condition ]
[ [ START WITH expression ] CONNECT BY expression ]
[ GROUP BY ALL | expression [, ...] ]
[ HAVING condition ]
[ QUALIFY condition ]
[ { UNION | ALL | INTERSECT | EXCEPT | MINUS } query ]
[ ORDER BY expression [ ASC | DESC ] ]
[ LIMIT { number | ALL } ]
[ OFFSET start ]
```

Pour en savoir plus sur les paramètres de cette commande, consultez SELECT.

Exemples

Sélectionnez toutes les lignes de la table EVENT et créez une table NEWEVENT :

```
select * into newevent from event;
```

Sélectionnez le résultat d'une requête d'agrégation dans une table temporaire appelée PROFITS :

```
select username, lastname, sum(pricepaid-commission) as profit
into temp table profits
from sales, users
where sales.sellerid=users.userid
group by 1, 2
order by 3 desc;
```

SET

Définit la valeur d'un paramètre de configuration du serveur. Utilisez la commande SET pour remplacer un paramètre pendant la durée de la séance ou de la transaction actuelle uniquement.

Utilisez la commande RESET pour rétablir un paramètre à sa valeur par défaut.

Vous pouvez modifier les paramètres de configuration du serveur de plusieurs manières. Pour plus d'informations, consultez Modification de la configuration du serveur.

Syntaxe

```
SET { [ SESSION | LOCAL ]
{ SEED | parameter_name } { TO | = }
{ value | 'value' | DEFAULT } |
SEED TO value }
```

L'instruction suivante définit la valeur d'une variable de contexte de session.

```
SET { [ SESSION | LOCAL ]
variable_name { TO | = }
{ value | 'value'  }
```

Paramètres

SESSION

Spécifie que le paramètre est valide pour la séance en cours. Valeur par défaut.

variable_name

Spécifie le nom de la variable contextuelle définie pour la session.

La convention de nommage est un nom en deux parties séparées par un point, par exemple identifier.identifier (identifiant.identifiant). Un seul séparateur de points est autorisé. Utilisez un identifiant qui respecte les règles d'identification standard pour Amazon Redshift. Pour obtenir plus d'informations à ce sujet, consultez Noms et identificateurs. Les identifiants délimités ne sont pas autorisés.

LOCAL

Spécifie que le paramètre est valide pour la transaction en cours.

SEED TO valeur

Définit une valeur initiale interne à utiliser par la fonction RANDOM pour la génération de nombres aléatoires.

SET SEED accepte une valeur numérique comprise entre 0 et 1, et la multiplie par (2^{31}-1) en vue de son utilisation avec la fonction Fonction RANDOM. Si vous utilisez SET SEED avant d'effectuer plusieurs appels RANDOM, RANDOM génère les nombres selon une séquence prévisible.

nom_paramètre

Nom du paramètre à définir. Consultez Modification de la configuration du serveur pour obtenir des informations sur les paramètres.

valeur

Nouvelle valeur de paramètre. Utilisez des guillemets simples pour définir la valeur d'une chaîne spécifique. Si vous utilisez SET SEED, ce paramètre contient la valeur SEED.

DEFAULT

Définit le paramètre à la valeur par défaut.

Exemples

Modification d'un paramètre pour la séance en cours

L'exemple suivant définit datestyle :

```
set datestyle to 'SQL,DMY';
```

Définition d'un groupe de requêtes pour la gestion de la charge de travail

Si les groupes de requêtes sont répertoriés dans une définition de file d'attente dans le cadre de la configuration WLM du cluster, vous pouvez définir le paramètre QUERY_GROUP avec un nom de groupe de requêtes répertorié. Les requêtes suivantes sont affectées à la file d'attente des requêtes associée. Le paramètre QUERY_GROUP reste en vigueur pendant la durée de la séance ou jusqu'à ce qu'une commande RESET_QUERY_GROUP soit rencontrée.

Cet exemple exécute deux requêtes dans le cadre du groupe de requêtes 'priority', puis réinitialise le groupe de requêtes.

```
set query_group to 'priority';
select tbl, count(*)from stv_blocklist;
select query, elapsed, substring from svl_qlog order by query desc limit 5;
reset query_group;
```

Pour de plus amples informations, veuillez consulter Gestion de la charge de travail.

Modifier l'espace de noms d'identité par défaut pour la session

Un utilisateur de base de données peut définir`default_identity_namespace`. Cet exemple montre comment remplacer le `SET SESSION` paramètre pour la durée de la session en cours, puis afficher la nouvelle valeur du fournisseur d'identité. C'est le plus souvent utilisé lorsque vous utilisez un fournisseur d'identité avec Redshift et IAM Identity Center. Pour plus d'informations sur l'utilisation d'un fournisseur d'identité avec Redshift, consultez Connect Redshift à IAM Identity Center pour offrir aux utilisateurs une expérience d'authentification unique.

```
SET SESSION default_identity_namespace = 'MYCO';

SHOW default_identity_namespace;
```

Après avoir exécuté la commande, vous pouvez exécuter une instruction GRANT ou une instruction CREATE comme suit :

```
GRANT SELECT ON TABLE mytable TO alice;

GRANT UPDATE ON TABLE mytable TO salesrole;

CREATE USER bob password 'md50c983d1a624280812631c5389e60d48c';
```

Dans ce cas, la définition de l'espace de noms d'identité par défaut équivaut à préfixer chaque identité par l'espace de noms. Dans cet exemple, `alice` est remplacé par`MYCO:alice`. Pour plus d'informations sur les paramètres relatifs à la configuration de Redshift avec IAM Identity Center, consultez et. ALTER SYSTEM ALTER IDENTITY PROVIDER

Définition d’une étiquette pour un groupe de requêtes

Le paramètre QUERY_GROUP définit une étiquette pour une ou plusieurs requêtes exécutées dans la même séance après une commande SET. À son tour, cette étiquette est enregistrée lorsque les requêtes sont exécutées et peuvent être utilisées pour limiter les résultats renvoyés par les tables système STL_QUERY et STV_INFLIGHT et la vue SVL_QLOG.

```
show query_group;
query_group
-------------
unset
(1 row)

set query_group to '6 p.m.';


show query_group;
query_group
-------------
6 p.m.
(1 row)

select * from sales where salesid=500;
salesid | listid | sellerid | buyerid | eventid | dateid | ...
---------+--------+----------+---------+---------+--------+-----
500 |    504 |     3858 |    2123 |    5871 |   2052 | ...
(1 row)

reset query_group;

select query, trim(label) querygroup, pid, trim(querytxt) sql
```

```
from stl_query
where label ='6 p.m.';
query | querygroup |  pid  |                  sql
-------+------------+-------+---------------------------------------
57 | 6 p.m.     | 30711 | select * from sales where salesid=500;
(1 row)
```

Les étiquettes de groupe de requêtes sont un mécanisme utile pour isoler les requêtes ou groupes de requêtes exécutés dans le cadre de scripts. Vous n'avez pas besoin d'identifier et de suivre les requêtes en fonction de leurs étiquettes IDs ; vous pouvez les suivre en fonction de leurs étiquettes.

Définition d’une valeur de départ pour la génération de nombres aléatoires

L’exemple suivant utilise SET avec l’option SEED pour que la fonction RANDOM génère des nombres selon une séquence prévisible.

D’abord, renvoyez trois entiers RANDOM sans définir au préalable la valeur SEED :

```
select cast (random() * 100 as int);
int4
------
6
(1 row)

select cast (random() * 100 as int);
int4
------
68
(1 row)

select cast (random() * 100 as int);
int4
------
56
(1 row)
```

A présent, définissez la valeur SEED sur `.25` et renvoyez trois nombres RANDOM supplémentaires :

```
set seed to .25;

select cast (random() * 100 as int);
int4
------
```

```
21
(1 row)

select cast (random() * 100 as int);
int4
------
79
(1 row)

select cast (random() * 100 as int);
int4
------
12
(1 row)
```

Enfin, réinitialisez la valeur SEED sur `.25` et vérifiez que RANDOM renvoie les mêmes résultats que les trois appels précédents :

```
set seed to .25;

select cast (random() * 100 as int);
int4
------
21
(1 row)

select cast (random() * 100 as int);
int4
------
79
(1 row)

select cast (random() * 100 as int);
int4
------
12
(1 row)
```

L'exemple suivant définit une variable contextuelle personnalisée.

```
SET app_context.user_id TO 123;
SET app_context.user_id TO 'sample_variable_value';
```

SET SESSION AUTHORIZATION

Définit le nom d'utilisateur de la séance en cours.

Vous pouvez utiliser la commande SET SESSION AUTHORIZATION pour tester l'accès à la base de données en exécutant temporairement une séance ou une transaction en tant qu'utilisateur non privilégié, par exemple. Vous devez être un super-utilisateur de la base de données pour exécuter cette commande.

Syntaxe

```
SET [ LOCAL ] SESSION AUTHORIZATION { user_name | DEFAULT }
```

Paramètres

LOCAL

Spécifie que le paramètre est valide pour la transaction en cours. Spécifie que le paramètre est valide pour la séance en cours.

user_name

Nom de l'utilisateur à définir. Le nom d'utilisateur peut être écrit comme un identificateur ou une chaîne littérale.

DEFAULT

Définit le nom d'utilisateur de la séance avec la valeur par défaut.

Exemples

L'exemple suivant définit le nom d'utilisateur de la séance en cours comme étant `dwuser` :

```
SET SESSION AUTHORIZATION 'dwuser';
```

L'exemple suivant définit le nom d'utilisateur de la transaction en cours avec la valeur `dwuser` :

```
SET LOCAL SESSION AUTHORIZATION 'dwuser';
```

Cet exemple définit le nom d'utilisateur de la séance en cours avec le nom d'utilisateur par défaut :

```
SET SESSION AUTHORIZATION DEFAULT;
```

SET SESSION CHARACTERISTICS

Cette commande est obsolète.

MONTRER

Affiche la valeur actuelle d'un paramètre de configuration de serveur. Cette valeur peut être spécifique à la séance en cours si une commande SET est activée. Pour obtenir la liste des paramètres de configuration, consultez Référence de configuration.

Syntaxe

```
SHOW { parameter_name | ALL }
```

L'instruction suivante affiche la valeur actuelle d'une variable de contexte de session. Si la variable n'existe pas, Amazon Redshift renvoie une erreur.

```
SHOW variable_name
```

Paramètres

nom_paramètre

Affiche la valeur actuelle du paramètre spécifié.

ALL

Affiche les valeurs actuelles de tous les paramètres.

variable_name

Affiche la valeur actuelle de la variable spécifiée.

Exemples

L'exemple suivant affiche la valeur du paramètre query_group :

```
show query_group;
```

```
query_group

unset
(1 row)
```

L'exemple suivant affiche une liste de tous les paramètres et de leurs valeurs :

```
show all;
name        |   setting
--------------------+--------------
datestyle          | ISO, MDY
extra_float_digits | 0
query_group        | unset
search_path        | $user,public
statement_timeout  | 0
```

L'exemple suivant affiche la valeur actuelle de la variable spécifiée.

```
SHOW app_context.user_id;
```

SHOW COLUMNS

Affiche la liste de colonnes d'une table ainsi que certains attributs de colonne.

Chaque ligne de sortie se compose d'une liste séparée par des virgules indiquant le nom de la base de données, le nom du schéma, le nom de la table, le nom de la colonne, la position ordinale, la valeur par défaut de la colonne, le caractère nullable ou non, le type de données, la longueur maximale en caractères, la précision numérique et des remarques. Pour obtenir plus d'informations sur ces attributs, consultez SVV_ALL_COLUMNS.

Si le résultat de la commande SHOW COLUMNS compte plus de 10 000 colonnes, une erreur est renvoyée.

Autorisations requises

Pour afficher une colonne dans un tableau Amazon Redshift, l'utilisateur actuel doit satisfaire à l'un des critères suivants :

- Être un super-utilisateur.
- Soyez le propriétaire de la table.

- Le privilège USAGE a été accordé sur le schéma parent et le privilège SELECT sur la table ou le privilège SELECT sur la colonne.

Syntaxe

```
SHOW COLUMNS FROM TABLE database_name.schema_name.table_name [LIKE 'filter_pattern']
 [LIMIT row_limit ]
```

Paramètres

database_name

Nom de la base de données qui contient les tables à répertorier.

Pour afficher les tables dans un AWS Glue Data Catalog, spécifiez (`awsdatacatalog`) comme nom de base de données et assurez-vous que la configuration du système `data_catalog_auto_mount` est définie sur`true`. Pour de plus amples informations, veuillez consulter ALTER SYSTEM.

nom_schéma

Nom du schéma qui contient les tables à répertorier.

Pour afficher AWS Glue Data Catalog les tables, indiquez le nom AWS Glue de la base de données comme nom du schéma.

table_name

Nom de la table qui contient les colonnes à répertorier.

filter_pattern

Expression de caractères UTF-8 valide constituée d'un modèle à mettre en correspondance avec les noms de tables. L'option LIKE effectue une mise en correspondance sensible à la casse qui prend en charge les métacaractères de mise en correspondance de modèle suivants :

Métacaractère	Description
%	Met en correspondance une séquence de zéro ou plusieurs caractères.
_	Met en correspondance un seul caractère.

Si filter_pattern ne contient pas de métacaractères, le modèle représente uniquement la chaîne elle-même ; dans ce cas, LIKE a la même fonction que l'opérateur d'égalité.

row_limit

Nombre maximal de lignes à renvoyer. La valeur de row_limit peut aller de 0 à 10 000.

Exemples

L'exemple suivant montre les colonnes de la base de données Amazon Redshift nommée `sample_data_dev` qui se trouvent dans le schéma `tickit` et la table `event`.

```
SHOW COLUMNS FROM TABLE sample_data_dev.tickit.event;

 database_name     | schema_name | table_name | column_name | ordinal_position |
 column_default | is_nullable | data_type                    | character_maximum_length
 | numeric_precision | numeric_scale | remarks
-------------------+-------------+------------+-------------+------------------
+----------------+-------------+-----------------------------
+--------------------------+-------------------+------------------------
 sample_data_dev   | tickit      | event      | eventid     |                1 | NULL
         | NO          | integer                      | NULL                     |
           32 | 0             | NULL
 sample_data_dev   | tickit      | event      | eventid     |                2 | NULL
         | NO          | smallint                     | NULL                     |
           16 | 0             | NULL
 sample_data_dev   | tickit      | event      | eventid     |                3 | NULL
         | NO          | smallint                     | NULL                     |
           16 | 0             | NULL
 sample_data_dev   | tickit      | event      | eventid     |                4 | NULL
         | NO          | smallint                     | NULL                     |
           16 | 0             | NULL
 sample_data_dev   | tickit      | event      | eventid     |                5 | NULL
         | YES         | character varying            | 200                      |
         NULL | NULL          | NULL
 sample_data_dev   | tickit      | event      | eventid     |                6 | NULL
         | YES         | timestamp without time zo... | NULL                     |
         NULL | NULL          | NULL
```

L'exemple suivant montre les tables de la AWS Glue Data Catalog base de données nommée `awsdatacatalog` qui figurent dans le schéma `batman` et la table`nation`. La sortie est limitée à 2 lignes.

```
SHOW COLUMNS FROM TABLE awsdatacatalog.batman.nation LIMIT 2;

 database_name  | schema_name | table_name | column_name | ordinal_position
 | column_default | is_nullable | data_type | character_maximum_length |
 numeric_precision | remarks
----------------+-------------+------------+-------------+-----------------
+----------------+-------------+-----------+-------------------------
+-------------------+---------
 awsdatacatalog | batman      | nation     | n_nationkey |               1 |
      |             | integer   |                         |                 |
 awsdatacatalog | batman      | nation     | n_name      |               2 |
      |             | character |                         |                 |
```

SHOW EXTERNAL TABLE

Affiche la définition d'une table externe, y compris les attributs de table et les attributs de colonne. Vous pouvez utiliser la sortie de l'instruction SHOW EXTERNAL TABLE pour recréer la table.

Pour plus d'informations sur la création de tables externes, consultez CREATE EXTERNAL TABLE.

Syntaxe

```
SHOW EXTERNAL TABLE [external_database].external_schema.table_name [ PARTITION ]
```

Paramètres

external_database

Nom de la base de données externe associée. Ce paramètre est facultatif.

external_schema

Nom du schéma externe associé.

table_name

Nom de la table à afficher.

PARTITION

Affiche les instructions ALTER TABLE pour ajouter des partitions à la définition de table.

Exemples

Les exemples suivants sont basés sur une table externe définie comme suit :

```
CREATE EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned (
     csmallint smallint,
     cint int,
     cbigint bigint,
     cfloat float4,
     cdouble float8,
     cchar char(10),
     cvarchar varchar(255),
     cdecimal_small decimal(18,9),
     cdecimal_big decimal(30,15),
     ctimestamp TIMESTAMP,
     cboolean boolean,
     cstring varchar(16383)
)
PARTITIONED BY (cdate date, ctime TIMESTAMP)
STORED AS PARQUET
LOCATION 's3://amzn-s3-demo-bucket/alldatatypes_parquet_partitioned';
```

Voici un exemple de la commande SHOW EXTERNAL TABLE et de sortie pour la table `my_schema.alldatatypes_parquet_test_partitioned`.

```
SHOW EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned;
```

```
"CREATE EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned (
    csmallint smallint,
    cint int,
    cbigint bigint,
    cfloat float4,
    cdouble float8,
    cchar char(10),
    cvarchar varchar(255),
    cdecimal_small decimal(18,9),
    cdecimal_big decimal(30,15),
```

```
    ctimestamp timestamp,
    cboolean boolean,
    cstring varchar(16383)
)
PARTITIONED BY (cdate date, ctime timestamp)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://amzn-s3-demo-bucket/alldatatypes_parquet_partitioned';"
```

Vous trouverez ci-dessous un exemple de commande SHOW EXTERNAL TABLE et de sortie pour la même table, mais avec la base de données également spécifiée dans le paramètre.

```
SHOW EXTERNAL TABLE my_database.my_schema.alldatatypes_parquet_test_partitioned;
```

```
"CREATE EXTERNAL TABLE my_database.my_schema.alldatatypes_parquet_test_partitioned (
    csmallint smallint,
    cint int,
    cbigint bigint,
    cfloat float4,
    cdouble float8,
    cchar char(10),
    cvarchar varchar(255),
    cdecimal_small decimal(18,9),
    cdecimal_big decimal(30,15),
    ctimestamp timestamp,
    cboolean boolean,
    cstring varchar(16383)
)
PARTITIONED BY (cdate date, ctime timestamp)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://amzn-s3-demo-bucket/alldatatypes_parquet_partitioned';"
```

Voici un exemple de la commande SHOW EXTERNAL TABLE et de sortie lorsque le paramètre PARTITION est utilisé. La sortie contient des instructions ALTER TABLE permettant d'ajouter des partitions à la définition de table.

```
SHOW EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned PARTITION;
```

```
"CREATE EXTERNAL TABLE my_schema.alldatatypes_parquet_test_partitioned (
    csmallint smallint,
    cint int,
    cbigint bigint,
    cfloat float4,
    cdouble float8,
    cchar char(10),
    cvarchar varchar(255),
    cdecimal_small decimal(18,9),
    cdecimal_big decimal(30,15),
    ctimestamp timestamp,
    cboolean boolean,
    cstring varchar(16383)
)
PARTITIONED BY (cdate date)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://amzn-s3-demo-bucket/alldatatypes_parquet_partitioned';
ALTER TABLE my_schema.alldatatypes_parquet_test_partitioned ADD IF NOT
 EXISTS PARTITION (cdate='2021-01-01') LOCATION 's3://amzn-s3-demo-bucket/
alldatatypes_parquet_partitioned2/cdate=2021-01-01';
ALTER TABLE my_schema.alldatatypes_parquet_test_partitioned ADD IF NOT
 EXISTS PARTITION (cdate='2021-01-02') LOCATION 's3://amzn-s3-demo-bucket/
alldatatypes_parquet_partitioned2/cdate=2021-01-02';"
```

SHOW DATABASES

Affiche les bases de données d'un catalogue de données ou d'un entrepôt de données Amazon Redshift. SHOW DATABASES répertorie toutes les bases de données accessibles, telles que, dans l'entrepôt de données, les AWS Glue Data Catalog bases de données (awsdatacatalog), les bases de données de partage de données et les bases de données Lake Formation.

Autorisations requises

Toutes les bases de données sont visibles par les utilisateurs, sauf :

- Pour les bases de données créées à partir d'un partage de données avec des autorisations permettant d'être visibles, l'utilisateur actuel doit disposer de l'autorisation USAGE sur la base de données.

Syntaxe

Pour afficher les bases de données d'un entrepôt de données Amazon Redshift :

```
SHOW DATABASES
[ LIKE '<expression>' ]
[ LIMIT row_limit ]
```

Pour afficher les bases de données d'un catalogue de données :

```
SHOW DATABASES FROM DATA CATALOG
[ ACCOUNT  '<id1>', '<id2>', ... ]
[ LIKE '<expression>' ]
[ IAM_ROLE default | 'SESSION' | 'arn:aws:iam::<account-id>:role/<role-name>' ]
[ LIMIT row_limit ]
```

Paramètres

ACCOUNT '<id1>', '<id2>', ...

Les AWS Glue Data Catalog comptes à partir desquels répertorier les bases de données. L'omission de ce paramètre signifie qu'Amazon Redshift doit afficher les bases de données du compte qui possède le cluster.

LIKE '<expression>'

Filtre la liste des bases de données sur celles qui correspondent à l'expression que vous spécifiez. Ce paramètre prend en charge les modèles qui utilisent les caractères génériques % (pourcentage) et _ (trait de soulignement).

IAM_ROLE default | 'SESSION' | 'arn:aws:iam::<account-id>:role/<role-name>'

Si vous spécifiez un rôle IAM qui est associé au cluster pendant l'exécution de la commande SHOW DATABASES, Amazon Redshift utilise les informations d'identification du rôle lorsque vous exécutez des requêtes sur la base de données.

Si le mot clé `default` est spécifié, le rôle IAM défini par défaut et qui est associé au cluster est alors utilisé.

Utilisez `'SESSION'` si vous vous connectez à votre cluster Amazon Redshift à l'aide d'une identité fédérée et que vous accédez aux tables à partir de la base de données externe créée à l'aide de la commande the section called "CREATE DATABASE". Pour voir un exemple

d'utilisation d'une identité fédéré, consultez Utilisation d'une identité fédérée pour gérer l'accès d'Amazon Redshift aux ressources locales et aux tables externes Amazon Redshift Spectrum, qui explique comment configurer l'identité fédérée.

Utilisez l'Amazon Resource Name (ARN) d'un rôle IAM que votre cluster utilise pour l'authentification et l'autorisation. Au minimum, le rôle IAM doit être autorisé à exécuter une opération LIST sur le compartiment Amazon S3 devant être accessible et une opération GET sur les objets Amazon S3 contenus dans le compartiment. Pour en savoir plus sur les bases de données créées à partir de AWS Glue Data Catalog for datashares et à l'aide de IAM_ROLE, consultez la section Utilisation de partages de données gérés par Lake Formation en tant que consommateur.

Le code suivant montre la syntaxe de la chaîne de paramètre IAM_ROLE pour un seul ARN.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-name>'
```

Vous pouvez créer des chaînes de rôles pour permettre à votre cluster d'endosser un autre rôle IAM, y compris un rôle appartenant à un autre compte. Les chaînes ainsi créées peuvent inclure jusqu'à 10 rôles. Pour plus d'informations, consultez Créer des rôles IAM dans Amazon Redshift Spectrum.

Attachez à ce rôle IAM une politique d'autorisations IAM similaire à la suivante.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AccessSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-west-2:123456789012:secret:my-
rds-secret-VNenFy"
        },
        {
            "Sid": "VisualEditor1",
```

```
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour connaître les étapes à suivre afin de créer un rôle IAM à utiliser avec une requête fédérée, consultez Création d'un secret et d'un rôle IAM pour utiliser des requêtes fédérées.

> **Note**
>
> N'incluez pas d'espaces dans la liste des rôles chaînés.

L'exemple suivant montre la syntaxe d'une chaîne de trois rôles.

```
IAM_ROLE 'arn:aws:iam::<aws-account-id>:role/<role-1-name>,arn:aws:iam::<aws-
account-id>:role/<role-2-name>,arn:aws:iam::<aws-account-id>:role/<role-3-name>'
```

LIMITE row_limit

Clause pour LIMITER le nombre de lignes renvoyées. Où row_limit est le nombre maximum de lignes à renvoyer. La valeur de row_limit peut aller de 0 à 10 000.

Exemples

L'exemple suivant affiche toutes les bases de données du catalogue de données à partir de l'ID de compte 123456789012.

```
SHOW DATABASES FROM DATA CATALOG ACCOUNT '123456789012'

  catalog_id  | database_name |                        database_arn
 |     type     |                                            target_database
                            | location | parameters
--------------+---------------+-------------------------------------------------------
+--------------
```

```
+-----------------------------------------------------------------------------------------------------------
+----------+------------
 123456789012 |   database1   | arn:aws:glue:us-east-1:123456789012:database/database1
 | Data Catalog |
                             |          |
 123456789012 |   database2   | arn:aws:glue:us-east-1:123456789012:database/database2
 | Data Catalog | arn:aws:redshift:us-
east-1:123456789012:datashare:035c45ea-61ce-86f0-8b75-19ac6102c3b7/database2 |
 |
```

Les exemples suivants montrent comment afficher toutes les bases de données du catalogue de données à partir de l'ID de compte 123456789012 en utilisant les informations d'identification d'un rôle IAM.

```
SHOW DATABASES FROM DATA CATALOG ACCOUNT '123456789012' IAM_ROLE default;
```

```
SHOW DATABASES FROM DATA CATALOG ACCOUNT '123456789012' IAM_ROLE <iam-role-arn>;
```

L'exemple suivant affiche toutes les bases de données de l'entrepôt de données Amazon Redshift connecté.

```
SHOW DATABASES

database_name  | database_owner | database_type        | database_acl | parameters |
 database_isolation_level
---------------+----------------+----------------------+--------------+------------
+--------------------
awsdatacatalog | 1              | auto mounted catalog | NULL         | UNKNOWN    |
 UNKNOWN
dev            | 1              | local                | NULL         | NULL       |
 Snapshot Isolation
```

AFFICHER LES SUBVENTIONS

Affiche les autorisations pour un utilisateur, un rôle ou un objet. L'objet peut être une base de données, un schéma, une table ou une fonction. Lorsque vous spécifiez un objet, tel qu'un tableau ou une fonction, vous devez le qualifier à l'aide d'une notation en deux ou trois parties. Par exemple, `schema_name.table_name` ou `database_name.schema_name.table_name`.

Syntaxe

Voici la syntaxe permettant d'afficher les autorisations accordées à un objet. Notez que la deuxième méthode de spécification d'une fonction n'est valable que pour les schémas externes et les bases de données créés à partir d'un partage de données.

```
SHOW GRANTS ON
{
 DATABASE database_name |
 FUNCTION {database_name.schema_name.function_name | schema_name.function_name }
 ( [ [ argname ] argtype [, ...] ] ) |
 FUNCTION {database_name.schema_name.function_name | schema_name.function_name } |
 SCHEMA {database_name.schema_name | schema_name} |
 { TABLE {database_name.schema_name.table_name | schema_name.table_name} | table_name }
}
[FOR {username | ROLE role_name | PUBLIC}]
[LIMIT row_limit]
```

Voici la syntaxe permettant d'afficher les autorisations accordées à un utilisateur ou à un rôle.

```
SHOW GRANTS FOR
{username | ROLE role_name}
[LIMIT row_limit]
```

Paramètres

database_name

Nom de la base de données sur laquelle afficher les autorisations.

function_name

Nom de la fonction sur laquelle afficher les subventions.

nom_schéma

Nom du schéma sur lequel les subventions doivent être affichées.

table_name

Nom du tableau sur lequel les subventions doivent être affichées.

POUR le nom d'utilisateur

Indique l'affichage des autorisations accordées à un utilisateur.

POUR ROLE role_name

Indique l'affichage des subventions pour un rôle.

POUR LE PUBLIC

Indique l'affichage des subventions pour PUBLIC.

row_limit

Nombre maximal de lignes à renvoyer. La valeur de row_limit peut aller de 0 à 10 000.

Exemples

L'exemple suivant affiche toutes les autorisations sur une base de données nomméedev.

```
SHOW GRANTS ON DATABASE dev;

 database_name | privilege_type | identity_id | identity_name | identity_type |
 admin_option | privilege_scope
---------------+----------------+-------------+---------------+---------------
+--------------+-----------------
 dev           | TRUNCATE       |         101 | alice         | user          | f
      | TABLES
 dev           | DROP           |         101 | alice         | user          | f
      | TABLES
 dev           | INSERT         |         101 | alice         | user          | f
      | TABLES
 dev           | ALTER          |         101 | alice         | user          | f
      | TABLES
 dev           | TEMP           |           0 | public        | public        | f
      | DATABASE
 dev           | DELETE         |         101 | alice         | user          | f
      | TABLES
 dev           | SELECT         |         101 | alice         | user          | f
      | TABLES
 dev           | UPDATE         |         101 | alice         | user          | f
      | TABLES
 dev           | REFERENCES     |         101 | alice         | user          | f
      | TABLES
(9 rows)
```

La commande suivante affiche toutes les autorisations sur un schéma nomméedemo.

```
SHOW GRANTS ON SCHEMA demo;

 schema_name | object_name | object_type | privilege_type | identity_id | identity_name
 | identity_type | admin_option | privilege_scope
-------------+-------------+-------------+----------------+-------------
+---------------+---------------+--------------+-----------------
 demo        | demo        | SCHEMA      | ALTER          |         101 | alice
 | user          | f             | SCHEMA
 demo        | demo        | SCHEMA      | DROP           |         101 | alice
 | user          | f             | SCHEMA
 demo        | demo        | SCHEMA      | USAGE          |         101 | alice
 | user          | f             | SCHEMA
 demo        | demo        | SCHEMA      | CREATE         |         101 | alice
 | user          | f             | SCHEMA
(4 rows)
```

La commande suivante affiche toutes les autorisations accordées à un utilisateur nomméalice.

```
SHOW GRANTS FOR alice;

 database_name | schema_name | object_name | object_type | privilege_type | identity_id
 | identity_name | identity_type | privilege_scope
---------------+-------------+-------------+-------------+----------------
+-------------+---------------+---------------+-----------------
 dev           |             |             | DATABASE    | INSERT         |         101
 | alice         | user          | TABLES
 dev           |             |             | DATABASE    | SELECT         |         101
 | alice         | user          | TABLES
 dev           |             |             | DATABASE    | UPDATE         |         101
 | alice         | user          | TABLES
 dev           |             |             | DATABASE    | DELETE         |         101
 | alice         | user          | TABLES
 dev           |             |             | DATABASE    | REFERENCES     |         101
 | alice         | user          | TABLES
 dev           |             |             | DATABASE    | DROP           |         101
 | alice         | user          | TABLES
 dev           |             |             | DATABASE    | TRUNCATE       |         101
 | alice         | user          | TABLES
 dev           |             |             | DATABASE    | ALTER          |         101
 | alice         | user          | TABLES
 dev           | public      | t1          | TABLE       | INSERT         |         101
 | alice         | user          | TABLE
```

```
 dev           | public      | t1          | TABLE       | SELECT         |         101
 | alice         | user          | TABLE
 dev           | public      | t1          | TABLE       | UPDATE         |         101
 | alice         | user          | TABLE
 dev           | public      | t1          | TABLE       | DELETE         |         101
 | alice         | user          | TABLE
 dev           | public      | t1          | TABLE       | REFERENCES     |         101
 | alice         | user          | TABLE
 dev           | public      | t1          | TABLE       | DROP           |         101
 | alice         | user          | TABLE
 dev           | public      | t1          | TABLE       | TRUNCATE       |         101
 | alice         | user          | TABLE
 dev           | public      | t1          | TABLE       | ALTER          |         101
 | alice         | user          | TABLE
 dev           | demo        |             | SCHEMA      | USAGE          |         101
 | alice         | user          | SCHEMA
 dev           | demo        |             | SCHEMA      | CREATE         |         101
 | alice         | user          | SCHEMA
 dev           | demo        |             | SCHEMA      | DROP           |         101
 | alice         | user          | SCHEMA
 dev           | demo        |             | SCHEMA      | ALTER          |         101
 | alice         | user          | SCHEMA
(20 rows)
```

La commande suivante affiche toutes les autorisations sur une table nommée en l'honneur `t3` d'un utilisateur nomméalice. Notez que vous pouvez utiliser une notation en deux ou en trois parties pour spécifier le nom de la table.

```
SHOW GRANTS ON TABLE demo_db.demo_schema.t3 FOR alice;
 schema_name | object_name | object_type | privilege_type | identity_id | identity_name
 | identity_type | admin_option | privilege_scope
-------------+-------------+-------------+----------------+-------------
+---------------+---------------+--------------+-----------------
 demo_schema | t3          | TABLE       | ALTER          |         100 | alice
 | user          | f             | TABLE
 demo_schema | t3          | TABLE       | TRUNCATE       |         100 | alice
 | user          | f             | TABLE
 demo_schema | t3          | TABLE       | DROP           |         100 | alice
 | user          | f             | TABLE
 demo_schema | t3          | TABLE       | SELECT         |         100 | alice
 | user          | f             | TABLE
```

```
 demo_schema | t3          | TABLE       | INSERT         |         100 | alice
 | user          | f            | TABLE
 demo_schema | t3          | TABLE       | UPDATE         |         100 | alice
 | user          | f            | TABLE
 demo_schema | t3          | TABLE       | DELETE         |         100 | alice
 | user          | f            | TABLE
 demo_schema | t3          | TABLE       | REFERENCES     |         100 | alice
 | user          | f            | TABLE
(8 rows)

SHOW GRANTS ON TABLE demo_schema.t3 FOR alice;
 schema_name | object_name | object_type | privilege_type | identity_id | identity_name
 | identity_type | admin_option | privilege_scope
-------------+-------------+-------------+----------------+-------------
+---------------+---------------+--------------+-----------------
 demo_schema | t3          | TABLE       | ALTER          |         100 | alice
 | user          | f            | TABLE
 demo_schema | t3          | TABLE       | TRUNCATE       |         100 | alice
 | user          | f            | TABLE
 demo_schema | t3          | TABLE       | DROP           |         100 | alice
 | user          | f            | TABLE
 demo_schema | t3          | TABLE       | SELECT         |         100 | alice
 | user          | f            | TABLE
 demo_schema | t3          | TABLE       | INSERT         |         100 | alice
 | user          | f            | TABLE
 demo_schema | t3          | TABLE       | UPDATE         |         100 | alice
 | user          | f            | TABLE
 demo_schema | t3          | TABLE       | DELETE         |         100 | alice
 | user          | f            | TABLE
 demo_schema | t3          | TABLE       | REFERENCES     |         100 | alice
 | user          | f            | TABLE
(8 rows)
```

L'exemple suivant montre toutes les autorisations dans une table nomméet4. Notez les différentes manières de spécifier le nom de la table.

```
SHOW GRANTS ON t4;
 schema_name | object_name | object_type | privilege_type | identity_id | identity_name
 | identity_type | admin_option | privilege_scope
-------------+-------------+-------------+----------------+-------------
+---------------+---------------+--------------+-----------------
 public      | t4          | TABLE       | ALTER          |         100 | alice
 | user          | f            | TABLE
```

```
 public      | t4          | TABLE       | TRUNCATE       |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | DROP           |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | SELECT         |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | INSERT         |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | UPDATE         |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | DELETE         |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | REFERENCES     |         100 | alice
 | user          | f            | TABLE
(8 rows)

SHOW GRANTS ON TABLE public.t4;
 schema_name | object_name | object_type | privilege_type | identity_id | identity_name
 | identity_type | admin_option | privilege_scope
-------------+-------------+-------------+----------------+-------------
+---------------+---------------+--------------+-----------------
 public      | t4          | TABLE       | ALTER          |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | TRUNCATE       |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | DROP           |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | SELECT         |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | INSERT         |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | UPDATE         |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | DELETE         |         100 | alice
 | user          | f            | TABLE
 public      | t4          | TABLE       | REFERENCES     |         100 | alice
 | user          | f            | TABLE
(8 rows)
```

SHOW MODEL

Affiche des informations utiles sur un modèle de machine learning, y compris son état, les paramètres utilisés pour le créer et la fonction de prédiction avec ses types d'arguments d'entrée. Vous pouvez

utiliser les informations issues de la commande SHOW MODEL pour recréer le modèle. Si les tables de base ont été modifiées, l'exécution de CREATE MODEL avec la même instruction SQL entraîne un modèle différent. Les informations renvoyées par la commande SHOW MODEL sont différentes pour le propriétaire du modèle et pour un utilisateur disposant du privilège EXECUTE. La commande SHOW MODEL affiche différentes sorties lorsqu'un modèle est entraîné à partir d'Amazon Redshift ou lorsqu'il s'agit d'un modèle BYOM.

Syntaxe

```
SHOW MODEL ( ALL | model_name )
```

Paramètres

ALL

Renvoie tous les modèles que l'utilisateur peut utiliser et leurs schémas.

model_name

Nom du modèle. Le nom du modèle d'un schéma doit être unique.

Notes d'utilisation

La commande SHOW MODEL renvoie le résultat suivant :

- Nom du modèle.
- Schéma dans lequel le modèle a été créé.
- Propriétaire du modèle.
- Heure de création du modèle.
- Statut du modèle, tel que READY, TRAINING ou FAILED.
- Message de motif d'un modèle ayant échoué.
- Erreur de validation si le modèle a terminé l'entraînement.
- Coût estimé pour dériver le modèle pour une approche autre que BYOM. Seul le propriétaire du modèle peut afficher ces informations.
- Liste des paramètres spécifiés par l'utilisateur et de leurs valeurs, notamment :
  - La colonne TARGET spécifiée.

- Le type de modèle, AUTO ou XGBoost.
- Le type de problème, tel que REGRESSION, BINARY_CLASSIFICATION ou MULTICLASS_CLASSIFICATION Ce paramètre est spécifique à AUTO.
- Nom de la tâche de formation Amazon SageMaker AI ou de la tâche Amazon SageMaker AI Autopilot qui a créé le modèle. Vous pouvez utiliser ce nom de poste pour obtenir plus d'informations sur le modèle sur Amazon SageMaker AI.
- L'objectif, par exemple MSE, F1 ou Précision. Ce paramètre est spécifique à AUTO.
- Nom de la fonction créée.
- Le type d'inférence, locale ou distante.
- Les arguments d'entrée de la fonction de prédiction.
- Les types d'argument d'entrée de la fonction de prédiction pour les modèles autres que BYOM (Bring Your Own Model).
- Le type de renvoi de la fonction de prédiction. Ce paramètre est spécifique au modèle BYOM.
- Nom du point de terminaison Amazon SageMaker AI pour un modèle BYOM avec inférence à distance.
- Rôle IAM. Seul le propriétaire du modèle peut afficher cette information.
- Compartiment S3 utilisé. Seul le propriétaire du modèle peut afficher cette information.
- La AWS KMS clé, si elle a été fournie. Seul le propriétaire du modèle peut afficher cette information.
- La durée maximale d'exécution du modèle.
- Si le type de modèle n'est pas AUTO, Amazon Redshift affiche également la liste des hyperparamètres fournis et leurs valeurs.

Vous pouvez également afficher certaines des informations fournies par SHOW MODEL dans d'autres tables catalogue, telles que pg_proc. Amazon Redshift renvoie des informations sur la fonction de prédiction enregistrée dans la table catalogue pg_proc. Ces informations incluent les noms des arguments d'entrée et leurs types pour la fonction de prédiction. Amazon Redshift renvoie les mêmes informations dans la commande SHOW MODEL.

```
SELECT * FROM pg_proc WHERE proname ILIKE '%<function_name>%';
```

Exemples

L'exemple suivant illustre la sortie de la commande SHOW MODEL.

```
SHOW MODEL ALL;

Schema Name |  Model Name
------------+---------------
 public     | customer_churn
```

Le propriétaire de customer_churn peut voir la sortie suivante. Un utilisateur disposant uniquement du privilège EXECUTE ne peut pas voir le rôle IAM, le compartiment Amazon S3 et le coût estimé du modèle.

```
SHOW MODEL customer_churn;

       Key                  |           Value
----------------------------+---------------------------------
 Model Name                 | customer_churn
 Schema Name                | public
 Owner                      | 'owner'
 Creation Time              | Sat, 15.01.2000 14:45:20
 Model State                | READY
 validation:F1              | 0.855
 Estimated Cost             | 5.7
                            |
 TRAINING DATA:             |
 Table                      | customer_data
 Target Column              | CHURN
                            |
 PARAMETERS:                |
 Model Type                 | auto
 Problem Type               | binary_classification
 Objective                  | f1
 Function Name              | predict_churn
 Function Parameters        | age zip average_daily_spend average_daily_cases
 Function Parameter Types   | int int float float
 IAM Role                   | 'iam_role'
 KMS Key                    | 'kms_key'
 Max Runtime                | 36000
```

SHOW DATASHARES

Affiche les partages entrants et sortants d'un cluster à partir du même compte ou entre comptes. Si vous ne spécifiez pas de nom d'unité de partage des données, Amazon Redshift affiche toutes

les unités de partage des données dans toutes les bases de données du cluster. Les utilisateurs disposant des privilèges ALTER et SHARE peuvent voir les partages pour lesquels ils disposent de privilèges.

Syntaxe

```
SHOW DATASHARES [ LIKE 'namepattern' ]
```

Paramètres

LIKE

Clause facultative qui compare le modèle de nom spécifié à la description de l'unité de partage des données. Lorsque cette clause est utilisée, Amazon Redshift affiche uniquement les unités de partage des données dont les noms correspondent au modèle de nom spécifié.

namepattern

Nom de l'unité de partage des données interrogée ou une partie du nom à mettre en correspondance à l'aide de caractères génériques.

Exemples

L'exemple suivant montre comment afficher les partages entrants et sortants pour un cluster.

```
SHOW DATASHARES;
SHOW DATASHARES LIKE 'sales%';

share_name   | share_owner | source_database | consumer_database | share_type |
 createdate          | is_publicaccessible | share_acl | producer_account |
 producer_namespace
-------------+-------------+-----------------+-------------------+------------
+---------------------+---------------------+-----------+------------------
+--------------------------------------
'salesshare' | 100         | dev             |                   | outbound
 | 2020-12-09 01:22:54.| False               |           |   123456789012   |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
```

SHOW PROCEDURE

Affiche la définition d'une procédure stockée donnée, y compris sa signature. Vous pouvez utiliser la sortie d'une commande SHOW PROCEDURE pour recréer la procédure stockée.

Syntaxe

```
SHOW PROCEDURE sp_name [( [ [ argname ] [ argmode ] argtype [, ...] ] )]
```

Paramètres

sp_name

Nom de la procédure à afficher.

[argname] [argmode] argtype

Types d'arguments en entrée pour identifier la procédure stockée. Si vous le souhaitez, vous pouvez inclure l'ensemble des types de données d'argument, y compris les arguments OUT. Ce n'est pas obligatoire si le nom de la procédure stockée est unique (c'est-à-dire non surchargé).

Exemples

L'exemple suivant illustre la définition de la procédure `test_spl2`.

```
show procedure test_sp2(int, varchar);
                                        Stored Procedure Definition
------------------------------------------------------------------------------------------------
CREATE OR REPLACE PROCEDURE public.test_sp2(f1 integer, INOUT f2 character varying, OUT
 character varying)
LANGUAGE plpgsql
AS $_$
DECLARE
out_var alias for $3;
loop_var int;
BEGIN
IF f1 is null OR f2 is null THEN
RAISE EXCEPTION 'input cannot be null';
END IF;
CREATE TEMP TABLE etl(a int, b varchar);
FOR loop_var IN 1..f1 LOOP
insert into etl values (loop_var, f2);
```

```
f2 := f2 || '+' || f2;
END LOOP;
SELECT INTO out_var count(*) from etl;
END;
$_$

(1 row)
```

SHOW SCHEMAS

Affiche la liste des schémas d'une base de données ainsi que certains attributs de schéma.

Chaque ligne de sortie comprend le nom de la base de données, le nom du schéma, le propriétaire du schéma, le type du schéma, l'ACL du schéma, la base de données source et l'option de schéma. Pour obtenir plus d'informations sur ces attributs, consultez SVV_ALL_SCHEMAS.

Si le résultat de la commande SHOW SCHEMAS compte plus de 10 000 schémas, une erreur est renvoyée.

Autorisations requises

Pour afficher un schéma dans une table Amazon Redshift, l'utilisateur actuel doit satisfaire à l'un des critères suivants :

- Être un super-utilisateur.
- Soyez le propriétaire du schéma.
- Privilège USAGE accordé sur le schéma.

Syntaxe

```
SHOW SCHEMAS FROM DATABASE database_name [LIKE 'filter_pattern'] [LIMIT row_limit ]
```

Paramètres

database_name

Nom de la base de données qui contient les tables à répertorier.

Pour afficher les tables dans un AWS Glue Data Catalog, spécifiez (`awsdatacatalog`) comme nom de base de données et assurez-vous que la configuration du système

`data_catalog_auto_mount` est définie sur`true`. Pour de plus amples informations, veuillez consulter ALTER SYSTEM.

filter_pattern

Expression de caractères UTF-8 valide avec un modèle à mettre en correspondance avec des noms de schéma. L'option LIKE effectue une mise en correspondance sensible à la casse qui prend en charge les métacaractères de mise en correspondance de modèle suivants :

Métacaractère	Description
`%`	Met en correspondance une séquence de zéro ou plusieurs caractères.
`_`	Met en correspondance un seul caractère.

Si filter_pattern ne contient pas de métacaractères, le modèle représente uniquement la chaîne elle-même ; dans ce cas, LIKE a la même fonction que l'opérateur d'égalité.

row_limit

Nombre maximal de lignes à renvoyer. La valeur de row_limit peut aller de 0 à 10 000.

Exemples

L'exemple suivant montre les schémas de la base de données Amazon Redshift nommée `dev`.

```
SHOW SCHEMAS FROM DATABASE dev;

 database_name |     schema_name      | schema_owner | schema_type |         schema_acl
          | source_database | schema_option
---------------+----------------------+--------------+-------------
+----------------------------+-----------------+--------------
 dev           | pg_automv            |            1 | local       |
          |                 |
 dev           | pg_catalog           |            1 | local       | jpuser=UC/
jpuser~=U/jpuser  |                 |
 dev           | public               |            1 | local       | jpuser=UC/
jpuser~=UC/jpuser |                 |
 dev           | information_schema   |            1 | local       | jpuser=UC/
jpuser~=U/jpuser  |                 |
```

```
 dev           | schemad79cd6d93bf043 |           1 | local       |
         |                 |
```

L'exemple suivant montre les schémas de la AWS Glue Data Catalog base de données nomméeawsdatacatalog. Le nombre maximal de lignes de sortie est 5.

```
SHOW SCHEMAS FROM DATABASE awsdatacatalog LIMIT 5;

 database_name  |     schema_name      | schema_owner | schema_type | schema_acl |
 source_database | schema_option
----------------+----------------------+--------------+-------------+------------
+-----------------+---------------
 awsdatacatalog | 000_too_many_glue_db |              | EXTERNAL    |            |
            |
 awsdatacatalog | 123_default          |              | EXTERNAL    |            |
            |
 awsdatacatalog | adhoc                |              | EXTERNAL    |            |
            |
 awsdatacatalog | all_shapes_10mb      |              | EXTERNAL    |            |
            |
 awsdatacatalog | all_shapes_1g        |              | EXTERNAL    |            |
            |
```

SHOW TABLE

Affiche la définition d'une table, y compris les attributs de table, les limites de table, les attributs de colonne et les limites de colonne. Vous pouvez utiliser la sortie de l'instruction SHOW TABLE pour recréer la table.

Pour plus d'informations sur la création de tables, consultez CREATE TABLE.

Syntaxe

```
SHOW TABLE [schema_name.]table_name
```

Paramètres

nom_schéma

(Facultatif) Nom du schéma associé.

table_name

Nom de la table à afficher.

Exemples

Voici un exemple de la sortie SHOW TABLE pour la table `sales`.

```
show table sales;
```

```
CREATE TABLE public.sales (
salesid integer NOT NULL ENCODE az64,
listid integer NOT NULL ENCODE az64 distkey,
sellerid integer NOT NULL ENCODE az64,
buyerid integer NOT NULL ENCODE az64,
eventid integer NOT NULL ENCODE az64,
dateid smallint NOT NULL,
qtysold smallint NOT NULL ENCODE az64,
pricepaid numeric(8,2) ENCODE az64,
commission numeric(8,2) ENCODE az64,
saletime timestamp without time zone ENCODE az64
)
DISTSTYLE KEY SORTKEY ( dateid );
```

Voici un exemple de la sortie SHOW TABLE pour la table `category` dans le schéma `public`.

```
show table public.category;
```

```
CREATE TABLE public.category (
catid smallint NOT NULL distkey,
catgroup character varying(10) ENCODE lzo,
catname character varying(10) ENCODE lzo,
catdesc character varying(50) ENCODE lzo
) DISTSTYLE KEY SORTKEY ( catid );
```

L'exemple suivant crée une table `foo` avec une clé primaire.

```
create table foo(a int PRIMARY KEY, b int);
```

Les résultats SHOW TABLE affichent l'instruction create avec toutes les propriétés de la table `foo`.

```
show table foo;
```

```
CREATE TABLE public.foo ( a integer NOT NULL ENCODE az64, b integer ENCODE az64,
 PRIMARY KEY (a) ) DISTSTYLE AUTO;
```

SHOW TABLES

Affiche la liste des tables d'un schéma ainsi que certains attributs de table.

Chaque ligne de sortie comprend le nom de la base de données, le nom du schéma, le nom de la table, le type de la table, l'ACL de la table et des remarques. Pour obtenir plus d'informations sur ces attributs, consultez SVV_ALL_TABLES.

Si le résultat de la commande SHOW TABLES compte plus de 10 000 tables, une erreur est renvoyée.

Autorisations requises

Pour afficher une table dans un schéma Amazon Redshift, l'utilisateur actuel doit satisfaire à l'un des critères suivants :

- Être un super-utilisateur.
- Soyez le propriétaire de la table.
- Le privilège USAGE a été accordé sur le schéma parent et le privilège SELECT sur la table ou le privilège SELECT sur n'importe quelle colonne de la table.

Syntaxe

```
SHOW TABLES FROM SCHEMA database_name.schema_name [LIKE 'filter_pattern']
 [LIMIT row_limit ]
```

Paramètres

database_name

Nom de la base de données qui contient les tables à répertorier.

Pour afficher les tables dans un AWS Glue Data Catalog, spécifiez (`awsdatacatalog`) comme nom de base de données et assurez-vous que la configuration du système

`data_catalog_auto_mount` est définie sur`true`. Pour de plus amples informations, veuillez consulter ALTER SYSTEM.

nom_schéma

Nom du schéma qui contient les tables à répertorier.

Pour afficher AWS Glue Data Catalog les tables, indiquez le nom AWS Glue de la base de données comme nom du schéma.

filter_pattern

Expression de caractères UTF-8 valide constituée d'un modèle à mettre en correspondance avec les noms de tables. L'option LIKE effectue une mise en correspondance sensible à la casse qui prend en charge les métacaractères de mise en correspondance de modèle suivants :

Métacaractère	Description
`%`	Met en correspondance une séquence de zéro ou plusieurs caractères.
`_`	Met en correspondance un seul caractère.

Si filter_pattern ne contient pas de métacaractères, le modèle représente uniquement la chaîne elle-même ; dans ce cas, LIKE a la même fonction que l'opérateur d'égalité.

row_limit

Nombre maximal de lignes à renvoyer. La valeur de row_limit peut aller de 0 à 10 000.

Exemples

L'exemple suivant montre les tables de la base de données Amazon Redshift nommée `dev` qui se trouvent dans le schéma `public`.

```
SHOW TABLES FROM SCHEMA dev.public;

 database_name | schema_name | table_name | table_type | table_acl | remarks
---------------+-------------+------------+------------+-----------+---------
 dev           | public      | tb         | TABLE      |           |
 dev           | public      | tb2        | TABLE      |           |
```

```
 dev           | public      | tb3        | TABLE      |           |
```

L'exemple suivant montre les tables de la AWS Glue Data Catalog base de données nommée `awsdatacatalog` qui figurent dans le schéma`batman`.

```
SHOW TABLES FROM SCHEMA awsdatacatalog.batman;

 database_name  | schema_name |    table_name    | table_type | table_acl | remarks
----------------+-------------+------------------+------------+-----------+---------
 awsdatacatalog | batman      | nation           | EXTERNAL   |           |
 awsdatacatalog | batman      | part             | EXTERNAL   |           |
 awsdatacatalog | batman      | partsupp         | EXTERNAL   |           |
 awsdatacatalog | batman      | region           | EXTERNAL   |           |
 awsdatacatalog | batman      | supplier         | EXTERNAL   |           |
 awsdatacatalog | batman      | automount_nation | EXTERNAL   |           |
```

SHOW VIEW

Affiche la définition d'une vue, y compris pour les vues matérialisées et les vues à liaison tardive. Vous pouvez utiliser la sortie de l'instruction SHOW VIEW pour recréer la vue.

Syntaxe

```
SHOW VIEW [schema_name.]view_name
```

Paramètres

nom_schéma

(Facultatif) Nom du schéma associé.

view_name

Nom de la vue à afficher.

Exemples

Voici la définition de vue pour la vue `LA_Venues_v`.

```
create view LA_Venues_v as select * from venue where venuecity='Los Angeles';
```

Voici un exemple de la commande SHOW VIEW et de sa sortie pour la vue définie ci-dessus.

```
show view LA_Venues_v;
```

```
SELECT venue.venueid,
venue.venuename,
venue.venuecity,
venue.venuestate,
venue.venueseats
FROM venue WHERE ((venue.venuecity)::text = 'Los Angeles'::text);
```

Voici la définition de vue pour la vue `public.Sports_v` dans le schéma `public`.

```
create view public.Sports_v as select * from category where catgroup='Sports';
```

Voici un exemple de la commande SHOW VIEW et de sa sortie pour la vue définie ci-dessus.

```
show view public.Sports_v;
```

```
SELECT category.catid,
category.catgroup,
category.catname,
category.catdesc
FROM category WHERE ((category.catgroup)::text = 'Sports'::text);
```

START TRANSACTION

Synonyme de la fonction BEGIN.

Consultez BEGIN.

TRUNCATE

Supprime toutes les lignes d’une table sans faire une analyse de table : cette opération est une alternative plus rapide pour une opération DELETE non qualifiée. Pour exécuter une commande TRUNCATE, vous devez disposer de l'autorisation TRUNCATE pour la table, être le propriétaire de la table ou être un superutilisateur. Pour accorder les autorisations de tronquer une table, utilisez la commande GRANT.

TRUNCATE est beaucoup plus efficace que DELETE et ne requiert ni opération VACUUM ni opération ANALYZE. Cependant, sachez que TRUNCATE valide la transaction dans laquelle l'opération est exécutée.

Syntaxe

```
TRUNCATE [ TABLE ] table_name
```

La commande fonctionne également sur une vue matérialisée.

```
TRUNCATE materialized_view_name
```

Paramètres

TABLE

Mot-clé facultatif.

table_name

Table temporaire ou permanente. Seul le propriétaire de la table ou un super-utilisateur peut tronquer une table.

Vous pouvez tronquer quelque table que ce soit, y compris les tables référencées dans des contraintes de clé étrangère.

Vous n'avez pas besoin d'exécuter une opération VACUUM sur une table après l'avoir tronquée.

materialized_view_name

Vue matérialisée.

Vous pouvez tronquer une vue matérialisée utilisée pour Diffusion de l'ingestion vers une vue matérialisée.

Notes d'utilisation

La commande TRUNCATE valide la transaction dans laquelle elle est exécutée ; par conséquent, vous ne pouvez pas annuler une opération TRUNCATE et une commande TRUNCATE peut valider les autres opérations quand elle se valide elle-même.

Exemples

Utilisez la commande TRUNCATE pour supprimer toutes les lignes de la table CATEGORY :

```
truncate category;
```

Essayez d'annuler une opération TRUNCATE :

```
begin;

truncate date;

rollback;

select count(*) from date;
count
-------
0
(1 row)
```

La table DATE demeure vide après la commande ROLLBACK, car la commande TRUNCATE s'est validée automatiquement.

L'exemple suivant utilise la commande TRUNCATE pour supprimer toutes les lignes d'une vue matérialisée.

```
truncate my_materialized_view;
```

Elle supprime tous les enregistrements de la vue matérialisée et laisse la vue matérialisée et son schéma intacts. Dans la requête, le nom de la vue matérialisée est un exemple.

UNLOAD

Le chiffrement côté client pour les commandes COPY et UNLOAD ne sera plus ouvert aux nouveaux clients à compter du 30 avril 2025. Si vous avez utilisé le chiffrement côté client avec les commandes COPY et UNLOAD au cours des 12 mois précédant le 30 avril 2025, vous pouvez continuer à utiliser le chiffrement côté client avec les commandes COPY ou UNLOAD jusqu'au 30 avril 2026. Après le 30 avril 2026, vous ne pourrez plus utiliser le chiffrement côté client pour COPY et UNLOAD. Nous vous recommandons de passer au chiffrement côté serveur

pour COPY et UNLOAD dès que possible. Si vous utilisez déjà le chiffrement côté serveur pour COPY et UNLOAD, il n'y a aucun changement et vous pouvez continuer à l'utiliser sans modifier vos requêtes. Pour plus d'informations sur le chiffrement pour COPY et UNLOAD, consultez le paramètre ENCRYPTED ci-dessous.

Décharge le résultat d’une requête dans un ou plusieurs fichiers texte, JSON ou Apache Parquet dans Amazon S3, à l’aide d’un chiffrement côté serveur Amazon S3 (SSE-S3). Vous pouvez également spécifier le chiffrement côté serveur à l'aide d'une AWS Key Management Service clé (SSE-KMS).

Par défaut, le fichier déchargé est au format texte délimité par une barre verticale (|).

Vous pouvez gérer la taille des fichiers sur Amazon S3 et, par extension, le nombre de fichiers, en définissant le paramètre MAXFILESIZE. Assurez-vous que les plages d’adresses IP S3 sont ajoutées à votre liste des autorisations. Pour plus d’informations sur les plages d’adresses IP S3 requises, consultez Isolement de réseau.

Vous pouvez transférer les résultats d’une requête Amazon Redshift vers le lac de données Amazon S3 dans Apache Parquet, un format de stockage en colonnes ouvert et efficace dédié à l’analyse. Le format Parquet est jusqu’à deux fois plus rapide à décharger et consomme jusqu’à six fois mois de stockage dans Amazon S3, en comparaison avec les formats texte. Cela vous permet de sauvegarder la transformation et l’enrichissement des données que vous avez faits dans Amazon S3 dans votre lac de données Amazon S3 dans un format ouvert. Vous pouvez ensuite analyser vos données avec Redshift Spectrum et d'autres AWS services tels qu'Amazon Athena, Amazon EMR et Amazon AI. SageMaker

Pour en savoir plus sur l’utilisation de la commande UNLOAD et pour voir des exemples de scénarios, consultez Déchargement de données dans Amazon Redshift.

Privilèges et autorisations nécessaires

Pour que la commande UNLOAD aboutisse, il est nécessaire de disposer au moins du privilège SELECT sur les données de la base de données, ainsi que de l’autorisation d’écrire à l’emplacement Amazon S3. Pour plus d'informations sur les autorisations d'accès aux AWS ressources pour la commande UNLOAD, consultezAutorisations d’accès aux autres ressources AWS.

Pour appliquer des autorisations de moindre privilège, suivez ces recommandations afin de n'accorder des autorisations, selon les besoins, qu'à l'utilisateur exécutant la commande.

- L'utilisateur doit disposer du privilège SELECT sur les données. Pour plus d'informations sur la façon de limiter les privilèges de base de données, consultezGRANT.
- L'utilisateur doit être autorisé à assumer le rôle IAM pour écrire dans le compartiment Amazon S3 de votre Compte AWS. Pour restreindre l'accès à un utilisateur de base de données afin qu'il puisse assumer un rôle, consultez la section Restreindre l'accès aux rôles IAM dans le guide de gestion Amazon Redshift.
- L'utilisateur doit avoir accès au compartiment Amazon S3. Pour restreindre les autorisations à l'aide d'une politique de compartiment Amazon S3, consultez les politiques de compartiment pour Amazon S3 dans le guide de l'utilisateur d'Amazon Simple Storage Service.

Syntaxe

```
UNLOAD ('select-statement')
TO 's3://object-path/name-prefix'
authorization
[ option, ...]

where authorization is
IAM_ROLE { default | 'arn:aws:iam::<Compte AWS-id-1>:role/<role-
name>[,arn:aws:iam::<Compte AWS-id-2>:role/<role-name>][,...]' }

where option is
| [ FORMAT [ AS ] ] CSV | PARQUET | JSON
| PARTITION BY ( column_name [, ... ] ) [ INCLUDE ]
| MANIFEST [ VERBOSE ]
| HEADER
| DELIMITER [ AS ] 'delimiter-char'
| FIXEDWIDTH [ AS ] 'fixedwidth-spec'
| ENCRYPTED [ AUTO ]
| BZIP2
| GZIP
| ZSTD
| ADDQUOTES
| NULL [ AS ] 'null-string'
| ESCAPE
| ALLOWOVERWRITE
| CLEANPATH
| PARALLEL [ { ON | TRUE } | { OFF | FALSE } ]
| MAXFILESIZE [AS] max-size [ MB | GB ]
| ROWGROUPSIZE [AS] size [ MB | GB ]
```

```
| REGION [AS] 'aws-region' }
| EXTENSION 'extension-name'
```

Paramètres

('instruction_select')

Requête SELECT. Les résultats de la requête sont déchargés. Dans la plupart des cas, il est utile de décharger les données dans un ordre trié en spécifiant une clause ORDER BY dans la requête. Cette approche économise le temps nécessaire au tri des données lors de leur rechargement.

La requête doit être placée entre guillemets simples comme illustré ci-après :

```
('select * from venue order by venueid')
```

> **Note**
>
> Si votre requête contient des guillemets (par exemple pour insérer les valeurs littérales), placez le littéral entre deux ensembles de guillemets simples. Vous devez également joindre la requête entre guillemets simples :
>
> ```
> ('select * from venue where venuestate=''NV''')
> ```

TO 's3://object-path/name-prefix'

Chemin complet, incluant le nom du compartiment, vers l'emplacement dans Amazon S3 où Amazon Redshift écrit les objets de fichier de sortie, y compris le fichier manifeste si MANIFEST est spécifié. Les noms d'objet sont préfixés par name-prefix. Si vous utilisez `PARTITION BY`, une barre oblique (/) est automatiquement ajoutée à la fin de la valeur name-prefix si nécessaire. Pour plus de sécurité, UNLOAD se connecte à Amazon S3 via une connexion HTTPS. Par défaut, UNLOAD écrit un ou plusieurs fichiers par tranche. UNLOAD ajoute un numéro de tranche et un numéro de partie au préfixe du nom spécifié comme suit :

<object-path>/*<name-prefix><slice-number>*_part_*<part-number>*.

Si MANIFEST est spécifié, le fichier manifeste est écrit comme suit :

<object_path>/*<name_prefix>*manifest.

Si PARALLEL est défini sur OFF, les fichiers de données sont écrits comme suit :

<object_path>/*<name_prefix>*<part-number>.

UNLOAD crée automatiquement les fichiers chiffrés à l'aide du chiffrement côté serveur Amazon S3, dont le fichier manifeste si MANIFEST est utilisé. La commande COPY lit automatiquement les fichiers chiffrés côté serveur pendant l'opération de chargement. Vous pouvez télécharger en toute transparence les fichiers chiffrés côté serveur à partir de votre compartiment à l'aide de la console de gestion ou de l'API Amazon S3. Pour plus d'informations, consultez Protection des données à l'aide du chiffrement côté serveur.

> **Important**
>
> La commande REGION est obligatoire lorsque le compartiment Amazon S3 ne se trouve pas dans la même Région AWS que la base de données Amazon Redshift.

authorization

La commande UNLOAD a besoin de l'autorisation d'écrire des données dans Amazon S3. La commande UNLOAD utilise les mêmes paramètres que ceux utilisés par la commande COPY pour l'autorisation. Pour plus d'informations, consultez Paramètres d'autorisation dans la référence de syntaxe de la commande COPY.

IAM_ROLE {par défaut | 'arn:aws:iam : :role/ '*<Compte AWS-id-1><role-name>*

Utilisez le mot clé par défaut pour qu'Amazon Redshift utilise le rôle IAM défini comme rôle par défaut et associé au cluster lorsque la commande UNLOAD est exécutée.

Utilisez l'Amazon Resource Name (ARN) d'un rôle IAM que votre cluster utilise pour l'authentification et l'autorisation. Si vous spécifiez IAM_ROLE, vous ne pouvez pas utiliser ACCESS_KEY_ID et SECRET_ACCESS_KEY, SESSION_TOKEN ni CREDENTIALS. IAM_ROLE peut être chaîné. Pour en savoir plus, consultez Chaînage des rôles IAM dans le Guide de gestion Amazon Redshift.

[FORMAT [AS]] CSV | PARQUET | JSON

Mots-clés pour spécifier le format de déchargement qui remplace le format par défaut.

Lorsque CSV est utilisé, décharge vers un fichier texte au format CSV en utilisant une virgule (,) comme délimiteur par défaut. Si un champ contient des délimiteurs, des guillemets doubles, des sauts de ligne ou des retours chariot, le champ du fichier déchargé est placé entre des guillemets doubles. Un caractère de guillemets doubles dans un champ de données est échappé par un caractère de guillemets doubles supplémentaires. Quand aucune ligne n'est déchargée, Amazon Redshift peut écrire des objets Amazon S3 vides.

Lorsque PARQUET est utilisé, décharge dans un fichier au format Apache Parquet version 1.0. Par défaut, chaque groupe de lignes est compressé à l'aide de la compression SNAPPY. Pour plus d'informations sur le format Apache Parquet, voir Parquet.

En cas d'utilisation de JSON, le fichier est déchargé dans un fichier JSON dont chaque ligne contient un objet JSON, représentant un enregistrement complet dans le résultat de la requête. Amazon Redshift prend en charge l'écriture de JSON imbriqué lorsque le résultat de la requête contient des colonnes SUPER. Pour créer un objet JSON valide, le nom de chaque colonne de la requête doit être unique. Dans le fichier JSON, les valeurs booléennes sont déchargées en tant que `t` ou `f`, et les valeurs NULL sont déchargées en tant que `null`. Quand aucune ligne n'est déchargée, Amazon Redshift n'écrit pas d'objets Amazon S3.

Les mots-clés FORMAT et AS sont facultatifs. Vous ne pouvez pas utiliser le format CSV avec ESCAPE, FIXEDWIDTH ou ADDQUOTES. Vous ne pouvez pas utiliser PARQUET avec DELIMITER, FIXEDWIDTH, ADDQUOTES, ESCAPE, NULL AS, HEADER, GZIP ou ZSTD. BZIP2 PARQUET with ENCRYPTED n'est pris en charge qu'avec le chiffrement côté serveur à l'aide d'une AWS Key Management Service clé (SSE-KMS). Vous ne pouvez pas utiliser JSON avec DELIMITER, HEADER, FIXEDWIDTH, ADDQUOTES, ESCAPE ou NULL AS.

PARTITION BY (nom_colonne [, ...]) [INCLUDE]

Spécifie les clés de partition pour l'opération de déchargement. UNLOAD partitionne automatiquement les fichiers de sortie dans des dossiers de partition en fonction des valeurs de clé de partition, conformément à la convention Apache Hive. Par exemple, un fichier Parquet pour l'année de partition 2019 et le mois de septembre a le préfixe suivant : `s3://amzn-s3-demo-bucket/my_prefix/year=2019/month=September/000.parquet`.

La valeur de nom_colonne doit être une colonne dans les résultats de requête déchargés.

Si vous spécifiez PARTITION BY avec l'option Inclure, les colonnes de partition ne sont pas supprimées des fichiers déchargés.

Amazon Redshift ne prend pas en charge les littéraux de chaîne dans les clauses PARTITION BY.

MANIFEST [VERBOSE]

Crée un fichier manifeste qui répertorie explicitement les détails des fichiers de données créés par le processus UNLOAD. Le manifeste est un fichier texte au format JSON qui répertorie l'URL de chaque fichier écrit sur Amazon S3.

Si MANIFEST est spécifié avec l'option VERBOSE, le manifeste inclut les informations suivantes :

- Les noms des colonnes et les types de données, et pour les types de données CHAR, VARCHAR ou NUMERIC, les dimensions de chaque colonne. Pour les types de données CHAR et VARCHAR, la dimension est la longueur. Pour un type de données DECIMAL ou NUMERIC, les dimensions sont la précision et l'échelle.
- Le nombre de lignes déchargées dans chaque fichier. Si l'option HEADER est spécifiée, le nombre de lignes inclut la ligne d'en-tête.
- La taille de fichier totale de tous les fichiers déchargés et le nombre total de lignes déchargées dans tous les fichiers. Si l'option HEADER est spécifiée, le nombre de lignes inclut les lignes d'en-tête.
- L'auteur. L'auteur est toujours « Amazon Redshift ».

Vous pouvez spécifier VERBOSE uniquement après MANIFEST.

Le fichier manifeste est écrit sur le même préfixe de chemin Amazon S3 que les fichiers de déchargement au format `<object_path_prefix>manifest`. Par exemple, si UNLOAD spécifie le préfixe de chemin Amazon S3 « `s3://amzn-s3-demo-bucket/venue_` », l'emplacement du fichier manifeste est « `s3://amzn-s3-demo-bucket/venue_manifest` ».

HEADER

Ajoute une ligne d'en-tête contenant des noms de colonne au début de chaque fichier de sortie. Les options de transformation de texte, comme CSV, DELIMITER, ADDQUOTES et ESCAPE, s'appliquent également à la ligne d'en-tête. Vous ne pouvez pas utiliser HEADER avec FIXEDWIDTH.

DELIMITER AS 'caractère_délimiteur'

Spécifie un caractère ASCII unique utilisé pour séparer les champs du fichier de sortie, tel qu'une barre verticale (|), une virgule (,) ou une tabulation (\t). Le délimiteur par défaut pour les fichiers texte est un caractère de barre verticale. Le délimiteur par défaut pour les fichiers CSV est un caractère de virgule. Le mot-clé AS est facultatif. Vous ne pouvez pas utiliser DELIMITER avec FIXEDWIDTH. Si les données contiennent le caractère délimiteur, vous devez spécifier l'option

ESCAPE pour insérer une séquence d'échappement devant le délimiteur ou utiliser ADDQUOTES pour placer les données entre guillemets doubles. Vous pouvez également spécifier un délimiteur qui se ne trouve pas dans les données.

FIXEDWIDTH 'fixedwidth_spec'

Décharge les données dans un fichier où la largeur de chaque colonne est une longueur fixe, plutôt que séparée par un délimiteur. fixedwidth_spec est une chaîne qui spécifie le nombre de colonnes et leur largeur. Le mot-clé AS est facultatif. Comme FIXEDWIDTH ne tronque pas les données, la spécification de chaque colonne dans l'instruction UNLOAD doit être au moins aussi longue que la longueur de l'entrée la plus longue de cette colonne. Le format de fixedwidth_spec est présenté ci-dessous :

```
'colID1:colWidth1,colID2:colWidth2, ...'
```

Vous ne pouvez pas utiliser FIXEDWIDTH avec DELIMITER ou HEADER.

ENCRYPTED [AUTO]

Spécifie que les fichiers de sortie sur Amazon S3 sont chiffrés à l'aide du chiffrement côté serveur Amazon S3. Si MANIFEST est spécifié, le fichier manifeste est également chiffré. Pour de plus amples informations, veuillez consulter Déchargement de fichiers de données chiffrés. Si vous ne spécifiez pas le paramètre ENCRYPTED, UNLOAD crée automatiquement des fichiers chiffrés à l'aide du chiffrement côté serveur Amazon S3 avec des clés de chiffrement AWS gérées (SSE-S3).

Pour ENCRYPTED, vous souhaiterez peut-être décharger sur Amazon S3 en utilisant le chiffrement côté serveur avec une AWS KMS clé (SSE-KMS). Le cas échéant, utilisez le paramètre KMS_KEY_ID pour fournir l'ID de clé. Vous ne pouvez pas utiliser le paramètre the section called "Utilisation des informations d'identification" avec le paramètre KMS_KEY_ID. Si vous exécutez une commande UNLOAD pour les données à l'aide de KMS_KEY_ID, vous pouvez effectuer une opération COPY pour les mêmes données sans spécifier de clé.

Si ENCRYPTED AUTO est utilisée, la commande UNLOAD extrait la clé de AWS KMS chiffrement par défaut sur la propriété du compartiment Amazon S3 cible et chiffre les fichiers écrits sur Amazon S3 avec cette clé. AWS KMS Si le compartiment ne possède pas la clé de AWS KMS chiffrement par défaut, UNLOAD crée automatiquement des fichiers chiffrés à l'aide du chiffrement côté serveur Amazon Redshift AWS avec des clés de chiffrement gérées (SSE-S3). Vous ne pouvez pas utiliser cette option avec KMS_KEY_ID, MASTER_SYMMETRIC_KEY ou CREDENTIALS contenant master_symmetric_key.

KMS_KEY_ID 'id_clé'

Spécifie l'ID de clé d'une clé AWS Key Management Service (AWS KMS) à utiliser pour chiffrer les fichiers de données sur Amazon S3. Pour plus d'informations, voir Qu'est-ce que c'est AWS Key Management Service ? Si vous spécifiez KMS_KEY_ID, vous devez également spécifier le paramètre ENCRYPTED. Si vous spécifiez KMS_KEY_ID, vous ne pouvez pas utiliser le paramètre CREDENTIALS. Utilisez the section called "Utilisation de IAM_ROLE" ou the section called "Utilisation de ACCESS_KEY_ID et SECRET_ACCESS_KEY" à la place.

BZIP2

Décharge les données sur un ou plusieurs fichiers compressé bzip2 par tranche. Chaque fichier résultant est ajouté avec une extension `.bz2`.

GZIP

Décharge les données sur un ou plusieurs fichiers compressé gzip par tranche. Chaque fichier résultant est ajouté avec une extension `.gz`.

ZSTD

Décharge les données sur un ou plusieurs fichiers compressé Zstandard par tranche. Chaque fichier résultant est ajouté avec une extension `.zst`.

ADDQUOTES

Place entre guillemets chaque champ de données déchargées, de telle sorte qu'Amazon Redshift puisse décharger les valeurs de données qui contiennent le délimiteur lui-même. Par exemple, si le délimiteur est une virgule, vous pouvez décharger et recharger les données suivantes avec succès :

```
"1","Hello, World"
```

Sans les guillemets ajoutés, la chaîne `Hello, World` serait analysée comme deux champs distincts.

Certains formats de sortie ne prennent pas en charge ADDQUOTES.

Si vous utilisez ADDQUOTES, vous devez spécifier REMOVEQUOTES dans la commande COPY si vous rechargez les données.

NULL AS ’chaîne_null’

Spécifie une chaîne qui représente une valeur null dans les fichiers de déchargement. Si cette option est utilisée, tous les fichiers de sortie contiennent la chaîne spécifiée à la place des valeurs null trouvées dans les données sélectionnées. Si cette option n’est pas spécifiée, les valeurs null sont déchargées en tant que :

- Chaînes de longueur nulle pour la sortie délimitée
- Chaînes d’espaces blancs pour la sortie de largeur fixe

Si une chaîne nulle a été spécifiée pour un déchargement de largeur fixe et que la largeur d’une colonne de sortie est inférieure à celle de la chaîne nulle, le comportement est le suivant :

- Un champ vide est généré pour les colonnes autres que les colonnes de caractères
- Une erreur est rapportée pour les colonnes de caractères

Contrairement à d’autres types de données où une chaîne définie par l’utilisateur représente une valeur nulle, Amazon Redshift exporte les colonnes de données SUPER en utilisant le format JSON et la représente comme une valeur nulle, comme déterminé par le format JSON. Par conséquent, les colonnes de données SUPER ignorent l'option NULL [AS] utilisée dans les commandes UNLOAD.

ESCAPE

Pour les colonnes CHAR et VARCHAR des fichiers de déchargement délimités, un caractère d’échappement (\) est placé devant toutes les occurrences des caractères suivants :

- Saut de ligne : \n
- Retour chariot: \r
- Délimiteur spécifié pour les données déchargées.
- Caractère d’échappement : \
- Guillemet : " ou ' (si ESCAPE et ADDQUOTES sont tous deux spécifiés dans la commande UNLOAD).

> **Important**
>
> Si vous avez chargé vos données à l’aide d’une commande COPY et de l’option ESCAPE, vous devez également spécifier l’option ESCAPE avec votre commande UNLOAD pour générer le fichier de sortie réciproque. De même, si vous utilisez UNLOAD avec l’option

ESCAPE, vous devez utiliser ESCAPE lorsque vous exécutez la commande COPY sur les mêmes données.

ALLOWOVERWRITE

Par défaut, UNLOAD échoue s'il trouve des fichiers qu'il pourrait remplacer. Si ALLOWOVERWRITE est spécifié, UNLOAD remplace les fichiers existants, y compris le fichier manifeste.

CLEANPATH

L'option CLEANPATH supprime les fichiers existants situés dans le chemin d'accès Amazon S3 spécifié dans la clause TO avant de décharger les fichiers à l'emplacement spécifié.

Si vous incluez la clause PARTITION BY, les fichiers existants sont supprimés uniquement des dossiers de partition pour recevoir les nouveaux fichiers générés par l'opération UNLOAD.

Vous devez bénéficier d'une autorisation `s3:DeleteObject` sur le compartiment Amazon S3. Pour obtenir des informations, consultez Politiques et autorisations dans Amazon S3 dans le Guide de l'utilisateur Amazon Simple Storage Service. Les fichiers que vous supprimez à l'aide de l'option CLEANPATH sont définitivement supprimés et ne peuvent pas être récupérés. Si la gestion des versions est activée dans le compartiment Amazon S3 cible, UNLOAD avec l'option CLEANPATH ne supprime pas les versions précédentes des fichiers.

Vous ne pouvez pas spécifier l'option CLEANPATH si vous spécifiez l'option ALLOWOVERWRITE.

PARALLEL

Par défaut, UNLOAD écrit les données en parallèle dans plusieurs fichiers, selon le nombre de tranches du cluster. L'option par défaut est ON ou TRUE. Si PARALLEL a la valeur OFF ou FALSE, UNLOAD écrit en série dans un ou plusieurs fichiers de données, triés de manière absolue selon la clause ORDER BY, si elle est utilisée. La taille maximale d'un fichier de données est de 6,2 Go. Ainsi, par exemple, si vous déchargez 13,4 Go de données, UNLOAD crée les trois fichiers suivants.

```
s3://amzn-s3-demo-bucket/key000    6.2 GB
s3://amzn-s3-demo-bucket/key001    6.2 GB
s3://amzn-s3-demo-bucket/key002    1.0 GB
```

> **Note**
>
> La commande UNLOAD est conçue pour utiliser le traitement parallèle. Nous vous recommandons de laisser PARALLEL activé dans la plupart des cas, surtout si les fichiers sont utilisés pour charger les tables à l'aide d'une commande COPY.

MAXFILESIZE [AS] max-size [Mo | Go]

Spécifie la taille maximale des fichiers créés par UNLOAD dans Amazon S3. Spécifiez une valeur décimale comprise entre 5 Mo et 6,2 Go. Le mot-clé AS est facultatif. L'unité par défaut est les Mo. Si MAXFILESIZE n'est pas spécifié, la taille maximale de fichier par défaut est de 6,2 Go. La taille du fichier manifeste, s'il est utilisé, n'est pas affectée par MAXFILESIZE.

ROWGROUPSIZE [AS] size [MB | GB]

Spécifie la taille des groupes de lignes. Le choix d'une taille supérieure peut réduire le nombre de groupes de lignes, réduisant ainsi la quantité de communication réseau. Spécifiez une valeur d'entier comprise entre 32 Mo et 128 Mo. Le mot-clé AS est facultatif. L'unité par défaut est les Mo.

Si ROWGROUPSIZE n'est pas spécifié, la taille par défaut est de 32 Mo. Pour utiliser ce paramètre, le format de stockage doit être Parquet et le type de nœud doit être ra3.4xlarge, ra3.16xlarge ou dc2.8xlarge.

REGION [AS] 'aws-region'

Spécifie l' Région AWS emplacement du compartiment Amazon S3 cible. La REGION est requise pour effectuer le DÉCHARGEMENT dans un compartiment Amazon S3 qui ne se trouve pas dans la Région AWS même base de données Amazon Redshift.

La valeur de aws_region doit correspondre à une AWS région répertoriée dans le tableau des régions et points de terminaison Amazon Redshift du. Références générales AWS

Par défaut, UNLOAD suppose que le compartiment Amazon S3 cible se trouve au même endroit Région AWS que la base de données Amazon Redshift.

EXTENSION 'extension-name'

Spécifie l'extension de fichier à ajouter aux noms des fichiers déchargés. Amazon Redshift n'effectuant aucune validation, vous devez vérifier que l'extension de fichier spécifiée est correcte.

Si vous spécifiez une méthode de compression sans indiquer d'extension, Amazon Redshift ajoute uniquement l'extension de la méthode de compression au nom de fichier. Si vous ne fournissez aucune extension et ne spécifiez pas de méthode de compression, Amazon Redshift n'ajoute rien au nom du fichier.

Notes d'utilisation

Utilisation d'ESCAPE pour toutes les opérations UNLOAD de texte délimité

Lorsque vous exécutez UNLOAD à l'aide d'un délimiteur, vos données peuvent inclure ce délimiteur ou l'un des caractères répertoriés dans la description de l'option ESCAPE. Dans ce cas, vous devez utiliser l'option ESCAPE avec l'instruction UNLOAD. Si vous n'utilisez pas l'option ESCAPE avec UNLOAD, les opérations COPY suivantes qui utilisent les données déchargées peuvent échouer.

> **Important**
>
> Nous vous recommandons vivement de toujours utiliser ESCAPE avec les deux instructions UNLOAD et COPY. Une exception s'applique si vous êtes certain que vos données ne contiennent pas de délimiteur ni d'autres caractères susceptibles d'avoir à faire l'objet d'un échappement.

Perte de précision pour la virgule flottante

Vous pouvez rencontrer une perte de précision pour les données en virgule flottante déchargées et rechargées avec succès.

Clause LIMIT

La requête SELECT ne peut pas utiliser une clause LIMIT dans l'instruction SELECT externe. Par exemple, l'instruction UNLOAD suivante échoue.

```
unload ('select * from venue limit 10')
to 's3://amzn-s3-demo-bucket/venue_pipe_' iam_role 'arn:aws:iam::0123456789012:role/
MyRedshiftRole';
```

À la place, utilisez une clause LIMIT imbriquée, comme dans l'exemple suivant.

```
unload ('select * from venue where venueid in
```

```
(select venueid from venue order by venueid desc limit 10)')
to 's3://amzn-s3-demo-bucket/venue_pipe_' iam_role 'arn:aws:iam::0123456789012:role/
MyRedshiftRole';
```

Vous pouvez aussi remplir une table à l'aide de SELECT…INTO ou de CREATE TABLE AS avec une clause LIMIT, puis procéder au déchargement à partir de cette table.

Déchargement d'une colonne avec le type de données GEOMETRY

Vous pouvez décharger les colonnes GEOMETRY au format texte ou CSV. Vous pouvez décharger les données GEOMETRY avec l'option `FIXEDWIDTH`. Les données sont déchargées au format hexadécimal du format EWKB. Si la taille des données EWKB est supérieure à 4 Mo, un avertissement est envoyé car les données ne pourront pas être chargées dans un table par la suite.

Déchargement du type de données HLLSKETCH

Vous pouvez décharger les colonnes HLLSKETCH au format texte ou CSV. Vous pouvez décharger les données HLLSKETCH avec l'option `FIXEDWIDTH`. Les données sont déchargées au format Base64 pour les HyperLogLog esquisses denses ou au format JSON pour les esquisses HyperLogLog éparses. Pour de plus amples informations, veuillez consulter HyperLogLog fonctions.

L'exemple suivant exporte une table contenant des colonnes HLLSKETCH dans un fichier.

```
CREATE TABLE a_table(an_int INT, b_int INT);
INSERT INTO a_table VALUES (1,1), (2,1), (3,1), (4,1), (1,2), (2,2), (3,2), (4,2),
 (5,2), (6,2);

CREATE TABLE hll_table (sketch HLLSKETCH);
INSERT INTO hll_table select hll_create_sketch(an_int) from a_table group by b_int;

UNLOAD ('select * from hll_table') TO 's3://amzn-s3-demo-bucket/unload/'
IAM_ROLE 'arn:aws:iam::0123456789012:role/MyRedshiftRole' NULL AS 'null' ALLOWOVERWRITE
 CSV;
```

Déchargement d'une colonne avec le type de données VARBYTE

Vous pouvez décharger les colonnes VARBYTE au format texte ou CSV. Les données sont déchargées sous forme hexadécimale. Vous ne pouvez pas décharger les données VARBYTE avec l'option `FIXEDWIDTH`. L'option `ADDQUOTES` de UNLOAD au format CSV n'est pas prise en charge. Une colonne VARBYTE ne peut pas être une colonne PARTITIONED BY.

Clause FORMAT AS PARQUET

Tenez comptes des points suivants lorsque vous utilisez FORMAT AS PARQUET :

- Une opération de déchargement vers Parquet n'utilise pas la compression au niveau du fichier. Chaque groupe de lignes est compressé avec SNAPPY.
- Si MAXFILESIZE n'est pas spécifié, la taille maximale de fichier par défaut est de 6,2 Go. Vous pouvez utiliser MAXFILESIZE pour spécifier une taille de fichier de 5 Mo à 6,2 Go. La taille réelle du fichier est approximative lorsque le fichier est écrit. Elle peut donc ne pas être exactement égale au nombre que vous spécifiez.

  Pour optimiser les performances d'analyse, Amazon Redshift essaie de créer des fichiers Parquet contenant des groupes de lignes de 32 Mo de taille égale. La valeur MAXFILESIZE que vous spécifiez est automatiquement arrondie au multiple le plus proche de 32 Mo. Par exemple, si vous spécifiez MAXFILESIZE 200 MB, chaque fichier Parquet déchargé est d'environ 192 Mo (groupe de lignes de 32 Mo x 6 = 192 Mo).
- Si une colonne utilise le format de données TIMESTAMPTZ, seules les valeurs d'horodatage sont déchargées. Les informations de fuseau horaire ne sont pas déchargées.
- Ne spécifiez pas de préfixes de nom de fichier commençant par des caractères de soulignement (_) ou des points (.). Redshift Spectrum traite les fichiers qui commencent par ces caractères comme des fichiers cachés et les ignore.

Clause PARTITION BY

Tenez comptes des points suivants lorsque vous utilisez PARTITION BY :

- Les colonnes de partition ne sont pas incluses dans le fichier de sortie.
- Assurez-vous d'inclure des colonnes de partition dans la requête SELECT utilisée dans l'instruction UNLOAD. Vous pouvez spécifier n'importe quel nombre de colonnes de partition dans la commande UNLOAD. Cependant, une limitation exige qu'au moins une colonne autre qu'une colonne de partition fasse partie du fichier.
- Si la valeur de clé de partition est null, Amazon Redshift décharge automatiquement ces données dans une partition par défaut appelée `partition_column=__HIVE_DEFAULT_PARTITION__`.
- La commande UNLOAD n'effectue pas d'appels vers un catalogue externe. Pour enregistrer vos nouvelles partitions dans le cadre de votre table externe existante, utilisez une commande distincte ALTER TABLE ... ADD PARTITION ... Vous pouvez également exécuter une commande CREATE EXTERNAL TABLE pour enregistrer les données déchargées en tant que nouvelle table externe.

Vous pouvez également utiliser un AWS Glue robot d'exploration pour remplir votre catalogue de données. Pour plus d'informations, consultez Définition des analyseurs dans le Guide du développeur AWS Glue .

- Si vous utilisez l'option MANIFEST, Amazon Redshift génère un seul fichier manifeste dans le dossier Amazon S3 racine.
- Les types de données de colonne que vous pouvez utiliser comme clé de partition sont SMALLINT, INTEGER, BIGINT, DECIMAL, REAL, BOOLEAN, CHAR, VARCHAR, DATE et TIMESTAMP.

Utilisation du privilège ASSUMEROLE pour accorder l'accès à un rôle IAM pour les opérations UNLOAD

Pour permettre à des utilisateurs et groupes spécifiques d'accéder à un rôle IAM pour les opérations UNLOAD, un super-utilisateur peut accorder le privilège ASSUMEROLE sur un rôle IAM aux utilisateurs et aux groupes. Pour obtenir des informations, consultez GRANT.

UNLOAD ne prend pas en charge les alias de point d'accès Amazon S3

Vous ne pouvez pas utiliser d'alias de point d'accès Amazon S3 avec la commande UNLOAD.

Exemples

Pour voir des exemples d'utilisation de la commande UNLOAD. consultez Exemples UNLOAD

Exemples UNLOAD

Ces exemples présentent différents paramètres de la commande UNLOAD. Les exemples de données TICKIT sont utilisés dans de nombreux exemples. Pour de plus amples informations, veuillez consulter Exemple de base de données.

> **Note**
>
> Ces exemples contiennent des sauts de ligne pour faciliter la lecture. N'incluez pas de sauts de ligne, ni d'espaces dans votre chaîne credentials-args.

Déchargement de VENUE sur un fichier délimité par une barre verticale (délimiteur par défaut)

L'exemple suivant décharge la table VENUE et écrit les données sur `s3://amzn-s3-demo-bucket/unload/`:

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Par défaut, UNLOAD écrit un ou plusieurs fichiers par tranche. En supposant que le cluster comporte deux nœuds avec deux tranches par nœud, l'exemple précédent crée les fichiers dans `amzn-s3-demo-bucket`:

```
unload/0000_part_00
unload/0001_part_00
unload/0002_part_00
unload/0003_part_00
```

Afin de mieux différencier les fichiers de sortie, vous pouvez inclure un préfixe dans l'emplacement. L'exemple suivant décharge la table VENUE et écrit les données sur `s3://amzn-s3-demo-bucket/unload/venue_pipe_`:

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Le résultat est les quatre fichiers du dossier `unload`, en supposant encore une fois quatre tranches.

```
venue_pipe_0000_part_00
venue_pipe_0001_part_00
venue_pipe_0002_part_00
venue_pipe_0003_part_00
```

Déchargement de la table LINEITEM dans des fichiers Parquet partitionnés

L'exemple suivant décharge la table LINEITEM au format Parquet, partitionné par la colonne `l_shipdate`.

```
unload ('select * from lineitem')
to 's3://amzn-s3-demo-bucket/lineitem/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
PARQUET
PARTITION BY (l_shipdate);
```

En supposant que quatre tranches sont utilisées, les fichiers Parquet résultants sont partitionnés dynamiquement dans divers dossiers.

```
s3://amzn-s3-demo-bucket/lineitem/l_shipdate=1992-01-02/0000_part_00.parquet
                                                0001_part_00.parquet
                                                0002_part_00.parquet
                                                0003_part_00.parquet
s3://amzn-s3-demo-bucket/lineitem/l_shipdate=1992-01-03/0000_part_00.parquet
                                                0001_part_00.parquet
                                                0002_part_00.parquet
                                                0003_part_00.parquet
s3://amzn-s3-demo-bucket/lineitem/l_shipdate=1992-01-04/0000_part_00.parquet
                                                0001_part_00.parquet
                                                0002_part_00.parquet
                                                0003_part_00.parquet
...
```

> **Note**
>
> Dans certains cas, la commande UNLOAD utilisait l'option INCLUDE comme indiqué dans l'instruction SQL suivante.
>
> ```
> unload ('select * from lineitem')
> to 's3://amzn-s3-demo-bucket/lineitem/'
> iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
> PARQUET
> PARTITION BY (l_shipdate) INCLUDE;
> ```
>
> Dans ces cas, la colonne `l_shipdate` se trouve également dans les données des fichiers Parquet. Sinon, les données de colonne `l_shipdate` ne se trouvent pas dans les fichiers Parquet.

Déchargez la table VENUE vers un fichier JSON

L'exemple suivant décharge la table VENUE et écrit les données en format JSON sur `s3://amzn-s3-demo-bucket/unload/`.

```
unload ('select * from venue')
```

```
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
JSON;
```

Vous trouverez ci-dessous des exemples de lignes de la table VENUE.

```
venueid | venuename                  | venuecity       | venuestate | venueseats
--------+----------------------------+-----------------+------------+-----------
      1 | Pinewood Racetrack         | Akron           | OH         | 0
      2 | Columbus "Crew" Stadium    | Columbus        | OH         | 0
      4 | Community, Ballpark, Arena | Kansas City     | KS         | 0
```

Après le déchargement au format JSON, le format du fichier est similaire au suivant.

```
{"venueid":1,"venuename":"Pinewood
 Racetrack","venuecity":"Akron","venuestate":"OH","venueseats":0}
{"venueid":2,"venuename":"Columbus \"Crew\" Stadium
 ","venuecity":"Columbus","venuestate":"OH","venueseats":0}
{"venueid":4,"venuename":"Community, Ballpark, Arena","venuecity":"Kansas
 City","venuestate":"KS","venueseats":0}
```

Décharger VENUE vers un fichier CSV

L'exemple suivant décharge la table VENUE et écrit les données en format CSV sur `s3://amzn-s3-demo-bucket/unload/`.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
CSV;
```

Supposons que la table VENUE contienne les lignes suivantes.

```
venueid | venuename                  | venuecity       | venuestate | venueseats
--------+----------------------------+-----------------+------------+-----------
      1 | Pinewood Racetrack         | Akron           | OH         | 0
      2 | Columbus "Crew" Stadium    | Columbus        | OH         | 0
      4 | Community, Ballpark, Arena | Kansas City     | KS         | 0
```

Le fichier de déchargement ressemble à ce qui suit.

```
1,Pinewood Racetrack,Akron,OH,0
2,"Columbus ""Crew"" Stadium",Columbus,OH,0
4,"Community, Ballpark, Arena",Kansas City,KS,0
```

Décharger VENUE dans un fichier CSV à l'aide d'un délimiteur

L'exemple suivant décharge la table VENUE et écrit les données au format CSV en utilisant le caractère de barre verticale (|) comme délimiteur. Le fichier déchargé est écrit dans `s3://amzn-s3-demo-bucket/unload/`. La table VENUE de cet exemple contient le caractère de barre verticale dans la valeur de la première ligne (`Pinewood Race|track`). Il le fait pour montrer que la valeur dans le résultat est entourée de guillemets doubles. Un guillemet double est échappé par un guillemet double, et le champ entier est entouré de guillemets doubles.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
CSV DELIMITER AS '|';
```

Supposons que la table VENUE contienne les lignes suivantes.

```
venueid | venuename                  | venuecity       | venuestate | venueseats
--------+----------------------------+-----------------+------------+-------------
      1 | Pinewood Race|track        | Akron           | OH         | 0
      2 | Columbus "Crew" Stadium    | Columbus        | OH         | 0
      4 | Community, Ballpark, Arena | Kansas City     | KS         | 0
```

Le fichier de déchargement ressemble à ce qui suit.

```
1|"Pinewood Race|track"|Akron|OH|0
2|"Columbus ""Crew"" Stadium"|Columbus|OH|0
4|Community, Ballpark, Arena|Kansas City|KS|0
```

Déchargement de VENUE avec un fichier manifeste

Pour créer un fichier manifeste, incluez l'option MANIFEST. L'exemple suivant décharge la table VENUE et écrit un fichier manifeste avec les fichiers de données sur s3://amzn-s3-demo-bucket/venue_pipe_ :

> **Important**
>
> Si vous déchargez les fichiers avec l'option MANIFEST, vous devez utiliser l'option MANIFEST avec la commande COPY lorsque vous chargez les fichiers. Si vous utilisez le même préfixe pour charger les fichiers et que vous ne spécifiez pas l'option MANIFEST, la commande COPY échoue, car elle suppose que le fichier manifeste est un fichier de données.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_pipe_' iam_role 'arn:aws:iam::0123456789012:role/
MyRedshiftRole'
manifest;
```

Le résultat est ces cinq fichiers :

```
s3://amzn-s3-demo-bucket/venue_pipe_0000_part_00
s3://amzn-s3-demo-bucket/venue_pipe_0001_part_00
s3://amzn-s3-demo-bucket/venue_pipe_0002_part_00
s3://amzn-s3-demo-bucket/venue_pipe_0003_part_00
s3://amzn-s3-demo-bucket/venue_pipe_manifest
```

Voici le contenu du fichier manifest.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0000_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0001_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0002_part_00"},
    {"url":"s3://amzn-s3-demo-bucket/tickit/venue_0003_part_00"}
  ]
}
```

Déchargement de VENUE avec MANIFEST VERBOSE

Lorsque vous spécifiez l'option MANIFEST VERBOSE, le fichier manifeste inclut les sections suivantes :

- La section `entries` répertorie le chemin Amazon S3, la taille de fichier et le nombre de lignes de chaque fichier.

- La section `schema` répertorie les noms de colonne, les types de données et la dimension de chaque colonne.
- La section `meta` montre la taille totale de fichier et le nombre total de lignes pour tous les fichiers.

L'exemple suivant décharge la table VENUE à l'aide de l'option MANIFEST VERBOSE.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload_venue_folder/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest verbose;
```

Voici le contenu du fichier manifest.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/venue_pipe_0000_part_00", "meta":
 { "content_length": 32295, "record_count": 10 }},
    {"url":"s3://amzn-s3-demo-bucket/venue_pipe_0001_part_00", "meta":
 { "content_length": 32771, "record_count": 20 }},
    {"url":"s3://amzn-s3-demo-bucket/venue_pipe_0002_part_00", "meta":
 { "content_length": 32302, "record_count": 10 }},
    {"url":"s3://amzn-s3-demo-bucket/venue_pipe_0003_part_00", "meta":
 { "content_length": 31810, "record_count": 15 }}
  ],
  "schema": {
    "elements": [
      {"name": "venueid", "type": { "base": "integer" }},
      {"name": "venuename", "type": { "base": "character varying", 25 }},
      {"name": "venuecity", "type": { "base": "character varying", 25 }},
      {"name": "venuestate", "type": { "base": "character varying", 25 }},
      {"name": "venueseats", "type": { "base": "character varying", 25 }}
    ]
  },
  "meta": {
    "content_length": 129178,
    "record_count": 55
  },
  "author": {
    "name": "Amazon Redshift",
    "version": "1.0.0"
  }
```

```
}
```

Déchargement de VENUE avec un en-tête

L'exemple suivant décharge VENUE avec une ligne d'en-tête.

```
unload ('select * from venue where venueseats > 75000')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
header
parallel off;
```

Voici le contenu du fichier de sortie avec une ligne d'en-tête.

```
venueid|venuename|venuecity|venuestate|venueseats
6|New York Giants Stadium|East Rutherford|NJ|80242
78|INVESCO Field|Denver|CO|76125
83|FedExField|Landover|MD|91704
79|Arrowhead Stadium|Kansas City|MO|79451
```

Déchargement de VENUE sur des fichiers plus petits

Par défaut, la taille maximale d'un fichier est de 6,2 Go. Si les données de déchargement sont supérieures à 6,2 Go, UNLOAD crée un nouveau fichier pour chaque segment de données de 6,2 Go. Pour créer des fichiers plus petits, incluez le paramètre MAXFILESIZE. En supposant que la taille des données de l'exemple précédent était de 20 Go, la commande UNLOAD suivante écrit 20 fichiers de 1 Go chacun.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
maxfilesize 1 gb;
```

Déchargement de VENUE en série

Pour décharger en série, spécifiez PARALLEL OFF. UNLOAD écrit ensuite un fichier à la fois, jusqu'à un maximum de 6,2 Go par fichier.

L'exemple suivant décharge la table VENUE et écrit les données en série sur `s3://amzn-s3-demo-bucket/unload/`.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/venue_serial_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off;
```

Le résultat est un fichier nommé venue_serial_000.

Si les données de déchargement sont supérieures à 6,2 Go, UNLOAD crée un nouveau fichier pour chaque segment de données de 6,2 Go. L'exemple suivant décharge la table LINEORDER et écrit les données en série sur `s3://amzn-s3-demo-bucket/unload/`.

```
unload ('select * from lineorder')
to 's3://amzn-s3-demo-bucket/unload/lineorder_serial_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
parallel off gzip;
```

Le résultat est la série de fichiers suivante.

```
lineorder_serial_0000.gz
lineorder_serial_0001.gz
lineorder_serial_0002.gz
lineorder_serial_0003.gz
```

Afin de mieux différencier les fichiers de sortie, vous pouvez inclure un préfixe dans l'emplacement. L'exemple suivant décharge la table VENUE et écrit les données sur `s3://amzn-s3-demo-bucket/venue_pipe_`:

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/unload/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Le résultat est les quatre fichiers du dossier `unload`, en supposant encore une fois quatre tranches.

```
venue_pipe_0000_part_00
venue_pipe_0001_part_00
venue_pipe_0002_part_00
venue_pipe_0003_part_00
```

Chargement de VENUE à partir des fichiers de déchargement

Pour charger une table à partir d'un ensemble de fichiers de déchargement, il suffit d'inverser le processus à l'aide d'une commande COPY. L'exemple suivant crée une table, LOADVENUE, et charge la table dans les fichiers de données créés dans l'exemple précédent.

```
create table loadvenue (like venue);

copy loadvenue from 's3://amzn-s3-demo-bucket/venue_pipe_' iam_role
 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Si vous avez utilisé l'option MANIFEST pour créer un fichier manifeste avec vos fichiers de déchargement, vous pouvez charger les données en utilisant le même fichier manifeste. Pour cela, vous utilisez une commande COPY avec l'option MANIFEST. L'exemple suivant charge les données à l'aide d'un fichier manifeste.

```
copy loadvenue
from 's3://amzn-s3-demo-bucket/venue_pipe_manifest' iam_role
 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest;
```

Déchargement de VENUE sur des fichiers chiffrés

L'exemple suivant décharge la table VENUE vers un ensemble de fichiers chiffrés à l'aide d'une AWS KMS clé. Si vous spécifiez un fichier manifeste avec l'option ENCRYPTED, le fichier manifeste est également chiffré. Pour plus d'informations, consultez Déchargement de fichiers de données chiffrés.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_encrypt_kms'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
kms_key_id '1234abcd-12ab-34cd-56ef-1234567890ab'
manifest
encrypted;
```

L'exemple suivant décharge la table VENUE sur un ensemble de fichiers chiffrés à l'aide d'une clé symétrique racine.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_encrypt_cmk'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
```

```
master_symmetric_key 'EXAMPLEMASTERKEYtkbjk/OpCwtYSx/M4/t7DMCDIK722'
encrypted;
```

Chargement de VENUE à partir de fichiers chiffrés

Pour charger les tables à partir d'un ensemble de fichiers qui ont été créés en utilisant UNLOAD avec l'option ENCRYPT, inversez le processus en utilisant une commande COPY. Avec cette commande, utilisez l'option ENCRYPTED et spécifiez la clé symétrique racine qui a été utilisée pour la commande UNLOAD. L'exemple suivant charge la tableau LOADVENUE à partir des fichiers de données chiffrés créés dans l'exemple précédent.

```
create table loadvenue (like venue);

copy loadvenue
from 's3://amzn-s3-demo-bucket/venue_encrypt_manifest'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
master_symmetric_key 'EXAMPLEMASTERKEYtkbjk/OpCwtYSx/M4/t7DMCDIK722'
manifest
encrypted;
```

Déchargement des données de VENUE dans un fichier délimité par les tabulations

```
unload ('select venueid, venuename, venueseats from venue')
to 's3://amzn-s3-demo-bucket/venue_tab_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter as '\t';
```

Les fichiers de données de sortie se présentent ainsi :

```
1 Toyota Park Bridgeview IL 0
2 Columbus Crew Stadium Columbus OH 0
3 RFK Stadium Washington DC 0
4 CommunityAmerica Ballpark Kansas City KS 0
5 Gillette Stadium Foxborough MA 68756
...
```

Déchargement de VENUE dans un fichier de données de largeur fixe

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_fw_'
```

```
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
fixedwidth as 'venueid:3,venuename:39,venuecity:16,venuestate:2,venueseats:6';
```

Les fichiers de données de sortie se présentent comme suit.

```
1  Toyota Park              Bridgeview  IL0
2  Columbus Crew Stadium    Columbus    OH0
3  RFK Stadium              Washington  DC0
4  CommunityAmerica BallparkKansas City KS0
5  Gillette Stadium         Foxborough  MA68756
...
```

Déchargement de VENUE sur un ensemble de fichiers compressés GZIP délimités par une tabulation

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_tab_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter as '\t'
gzip;
```

Déchargement de VENUE dans un fichier texte compressé par GZIP

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_tab_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
extension 'txt.gz'
gzip;
```

Déchargement des données qui contiennent un délimiteur

Cet exemple utilise l'option ADDQUOTES pour décharger les données délimitées par une virgule, où certains champs de données contiennent une virgule.

D'abord, créez une table qui contient des guillemets.

```
create table location (id int, location char(64));

insert into location values (1,'Phoenix, AZ'),(2,'San Diego, CA'),(3,'Chicago, IL');
```

Puis, déchargez les données à l'aide de l'option ADDQUOTES.

```
unload ('select id, location from location')
to 's3://amzn-s3-demo-bucket/location_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
delimiter ',' addquotes;
```

Les fichiers de données déchargés se présentent ainsi :

```
1,"Phoenix, AZ"
2,"San Diego, CA"
3,"Chicago, IL"
...
```

Décharger les résultats d'une requête de jointure

L'exemple suivant décharge les résultats d'une requête de jointure contenant une fonction de fenêtrage.

```
unload ('select venuecity, venuestate, caldate, pricepaid,
sum(pricepaid) over(partition by venuecity, venuestate
order by caldate rows between 3 preceding and 3 following) as winsum
from sales join date on sales.dateid=date.dateid
join event on event.eventid=sales.eventid
join venue on event.venueid=venue.venueid
order by 1,2')
to 's3://amzn-s3-demo-bucket/tickit/winsum'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Les fichiers de sortie se présentent ainsi :

```
Atlanta|GA|2008-01-04|363.00|1362.00
Atlanta|GA|2008-01-05|233.00|2030.00
Atlanta|GA|2008-01-06|310.00|3135.00
Atlanta|GA|2008-01-08|166.00|8338.00
Atlanta|GA|2008-01-11|268.00|7630.00
...
```

Décharger à l'aide de NULL AS

Par défaut, UNLOAD génère les valeurs null comme chaînes vides. Les exemples suivants montrent comment utiliser NULL AS pour remplacer les valeurs null par une chaîne de texte.

Pour ces exemples, nous ajoutons quelques valeurs null à la table VENUE.

```
update venue set venuestate = NULL
where venuecity = 'Cleveland';
```

Sélectionnez dans VENUE où VENUESTATE a la valeur null pour vérifier que les colonnes contiennent la valeur NULL.

```
select * from venue where venuestate is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
      22 | Quicken Loans Arena      | Cleveland |            |          0
     101 | Progressive Field        | Cleveland |            |      43345
      72 | Cleveland Browns Stadium | Cleveland |            |      73200
```

Maintenant, exécutez UNLOAD sur la table VENUE à l'aide de l'option NULL AS pour remplacer les valeurs null par la chaîne de caractères '`fred`'.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
null as 'fred';
```

L'exemple suivant du fichier de déchargement montre que les valeurs null ont été remplacées par `fred`. Il s'avère que certaines valeurs de VENUESEATS étaient également null et qu'elles ont été remplacées par `fred`. Même si le type de données pour VENUESEATS est entier, UNLOAD convertit les valeurs en texte dans les fichiers de déchargement, puis la commande COPY les reconvertit en type entier. Si vous déchargez dans un fichier à largeur fixe, la chaîne NULL AS ne doit pas être plus grande que la largeur du champ.

```
248|Charles Playhouse|Boston|MA|0
251|Paris Hotel|Las Vegas|NV|fred
258|Tropicana Hotel|Las Vegas|NV|fred
300|Kennedy Center Opera House|Washington|DC|0
306|Lyric Opera House|Baltimore|MD|0
308|Metropolitan Opera|New York City|NY|0
  5|Gillette Stadium|Foxborough|MA|5
 22|Quicken Loans Arena|Cleveland|fred|0
101|Progressive Field|Cleveland|fred|43345
```

```
...
```

Pour charger une table à partir des fichiers de déchargement, utilisez une commande COPY avec la même option NULL AS.

> **Note**
>
> Si vous essayez de charger les valeurs null dans une colonne définie comme NOT NULL, la commande COPY échoue.

```
create table loadvenuenulls (like venue);

copy loadvenuenulls from 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
null as 'fred';
```

Pour vérifier que les colonnes contiennent des chaînes null, et pas simplement des chaînes vides, sélectionnez LOADVENUENULLS et filtrez les valeurs null.

```
select * from loadvenuenulls where venuestate is null or venueseats is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
      72 | Cleveland Browns Stadium | Cleveland |            |      73200
     253 | Mirage Hotel             | Las Vegas | NV         |
     255 | Venetian Hotel           | Las Vegas | NV         |
      22 | Quicken Loans Arena      | Cleveland |            |          0
     101 | Progressive Field        | Cleveland |            |      43345
     251 | Paris Hotel              | Las Vegas | NV         |

...
```

Vous pouvez exécuter une opération UNLOAD sur une table qui contient des valeurs null en utilisant le comportement par défaut NULL AS, puis copier à nouveau les données dans une table en utilisant le comportement NULL AS ; cependant, tous les champs non numériques de la table cible contiennent des chaînes vides, pas des valeurs null. Par défaut, UNLOAD convertit les chaînes null en chaînes vides (espace blanc ou longueur égale à zéro). COPY convertit les chaînes vides en NULL pour les colonnes numériques, mais insère des chaînes vides dans les colonnes non

numériques. L'exemple suivant montre comment effectuer une opération UNLOAD suivi d'une opération COPY en utilisant le comportement par défaut NULL AS.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole' allowoverwrite;

truncate loadvenuenulls;
copy loadvenuenulls from 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole';
```

Dans ce cas, lorsque vous filtrez les valeurs null, seules les lignes où VENUESEATS contenait des valeurs null. Là où VENUESTATE contenait des valeurs null dans la table (VENUE), VENUESTATE dans la table cible (LOADVENUENULLS) contient des chaînes vides.

```
select * from loadvenuenulls where venuestate is null or venueseats is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
---------+--------------------------+-----------+------------+------------
     253 | Mirage Hotel             | Las Vegas | NV         |
     255 | Venetian Hotel           | Las Vegas | NV         |
     251 | Paris Hotel              | Las Vegas | NV         |
...
```

Pour charger des chaînes vides dans des colonnes non numériques comme NULL, incluez les options EMPTYASNULL ou BLANKSASNULL. Les deux peuvent être utilisés.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole' allowoverwrite;

truncate loadvenuenulls;
copy loadvenuenulls from 's3://amzn-s3-demo-bucket/nulls/'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole' EMPTYASNULL;
```

Pour vérifier que les colonnes contiennent NULL, et pas seulement des espaces ou des chaînes vides, effectuez une sélection à partir de LOADVENUENULLS et filtrez les valeurs null.

```
select * from loadvenuenulls where venuestate is null or venueseats is null;

 venueid |        venuename         | venuecity | venuestate | venueseats
```

```
---------+--------------------------+-----------+------------+------------
      72 | Cleveland Browns Stadium | Cleveland |            |      73200
     253 | Mirage Hotel             | Las Vegas | NV         |
     255 | Venetian Hotel           | Las Vegas | NV         |
      22 | Quicken Loans Arena      | Cleveland |            |          0
     101 | Progressive Field        | Cleveland |            |      43345
     251 | Paris Hotel              | Las Vegas | NV         |
     ...
```

Décharger à l'aide du paramètre ALLOWOVERWRITE

Par défaut, UNLOAD ne remplace pas les fichiers existants du compartiment de destination. Par exemple, si vous exécutez la même instruction UNLOAD deux fois sans modifier les fichiers du compartiment de destination, la deuxième opération UNLOAD échoue. Pour remplacer les fichiers existants, y compris le fichier manifeste, spécifiez l'option ALLOWOVERWRITE.

```
unload ('select * from venue')
to 's3://amzn-s3-demo-bucket/venue_pipe_'
iam_role 'arn:aws:iam::0123456789012:role/MyRedshiftRole'
manifest allowoverwrite;
```

Décharger la table EVENT à l'aide des paramètres PARALLEL et MANIFEST

Vous pouvez décharger (UNLOAD) une table en parallèle et générer un fichier manifeste. Les fichiers de données Amazon S3 sont tous créés au même niveau et les noms présentent le modèle `0000_part_00` en suffixe. Le fichier manifeste se trouve au même niveau de dossier que les fichiers de données et présentent le texte `manifest` en suffixe. Le code SQL suivant décharge la table EVENT et crée des fichiers avec le nom de base `parallel`

```
unload ('select * from mytickit1.event')
to 's3://amzn-s3-demo-bucket/parallel'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
parallel on
manifest;
```

La liste de fichiers Amazon S3 se présente comme suit.

```
 Name                        Last modified                         Size
 parallel0000_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 52.1 KB
 parallel0001_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 53.4 KB
```

```
 parallel0002_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 52.1 KB
 parallel0003_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 51.1 KB
 parallel0004_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 54.6 KB
 parallel0005_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 53.4 KB
 parallel0006_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 54.1 KB
 parallel0007_part_00 -   August 2, 2023, 14:54:39 (UTC-07:00) 55.9 KB
 parallelmanifest       -   August 2, 2023, 14:54:39 (UTC-07:00) 886.0 B
```

Le contenu du fichier `parallelmanifest` se présente comme suit.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/parallel0000_part_00", "meta": { "content_length":
 53316 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0001_part_00", "meta": { "content_length":
 54704 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0002_part_00", "meta": { "content_length":
 53326 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0003_part_00", "meta": { "content_length":
 52356 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0004_part_00", "meta": { "content_length":
 55933 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0005_part_00", "meta": { "content_length":
 54648 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0006_part_00", "meta": { "content_length":
 55436 }},
    {"url":"s3://amzn-s3-demo-bucket/parallel0007_part_00", "meta": { "content_length":
 57272 }}
  ]
}
```

Décharger la table EVENT à l'aide des PARALLEL OFFet MANIFEST

Vous pouvez décharger (UNLOAD) une table en série (PARALLEL OFF) et générer un fichier manifeste. Les fichiers de données Amazon S3 sont tous créés au même niveau et les noms présentent le modèle `0000` en suffixe. Le fichier manifeste se trouve au même niveau de dossier que les fichiers de données et présentent le texte `manifest` en suffixe.

```
unload ('select * from mytickit1.event')
to 's3://amzn-s3-demo-bucket/serial'
```

```
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
parallel off
manifest;
```

La liste de fichiers Amazon S3 se présente comme suit.

```
 Name                     Last modified                       Size
 serial0000           -   August 2, 2023, 15:54:39 (UTC-07:00) 426.7 KB
 serialmanifest       -   August 2, 2023, 15:54:39 (UTC-07:00) 120.0 B
```

Le contenu du fichier `serialmanifest` se présente comme suit.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/serial000", "meta": { "content_length": 436991 }}
  ]
}
```

Décharger la table EVENT à l'aide des paramètres PARTITION BY et MANIFEST

Vous pouvez décharger (UNLOAD) une table par partition et générer un fichier manifeste. Un dossier est créé dans Amazon S3 avec des dossiers de partition enfants, et les fichiers de données contenus dans les dossiers enfants ont un modèle de nom similaire à `0000_par_00`. Le fichier manifeste se trouve au même niveau de dossier que les dossiers enfants et se nomme `manifest`.

```
unload ('select * from mytickit1.event')
to 's3://amzn-s3-demo-bucket/partition'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
partition by (eventname)
manifest;
```

La liste de fichiers Amazon S3 se présente comme suit.

```
 Name                  Type     Last modified                     Size

 partition            Folder
```

Dans le dossier `partition` se trouvent les dossiers enfants avec le nom de la partition et le fichier manifeste. Le bas de la liste des dossiers contenus dans le dossier `partition` se présente comme suit.

```
Name                    Type      Last modified                          Size

...
eventname=Zucchero/     Folder
eventname=Zumanity/     Folder
eventname=ZZ Top/       Folder
manifest                -     August 2, 2023, 15:54:39 (UTC-07:00) 467.6 KB
```

Dans le dossier `eventname=Zucchero/` figurent les fichiers de données, comme ci-dessous.

```
Name               Last modified                         Size
0000_part_00 -   August 2, 2023, 15:59:19 (UTC-07:00) 70.0 B
0001_part_00 -   August 2, 2023, 15:59:16 (UTC-07:00) 106.0 B
0002_part_00 -   August 2, 2023, 15:59:15 (UTC-07:00) 70.0 B
0004_part_00 -   August 2, 2023, 15:59:17 (UTC-07:00) 141.0 B
0006_part_00 -   August 2, 2023, 15:59:16 (UTC-07:00) 35.0 B
0007_part_00 -   August 2, 2023, 15:59:19 (UTC-07:00) 108.0 B
```

Le bas du contenu du fichier `manifest` se présente comme suit.

```
{
  "entries": [
    ...
    {"url":"s3://amzn-s3-demo-bucket/partition/eventname=Zucchero/007_part_00", "meta":
 { "content_length": 108 }},
    {"url":"s3://amzn-s3-demo-bucket/partition/eventname=Zumanity/007_part_00", "meta":
 { "content_length": 72 }}
  ]
}
```

Décharger la table EVENT à l'aide des paramètres MAXFILESIZE, ROWGROUPSIZE et MANIFEST

Vous pouvez décharger (UNLOAD) une table en parallèle et générer un fichier manifeste. Les fichiers de données Amazon S3 sont tous créés au même niveau et les noms présentent le

modèle `0000_part_00` en suffixe. Les fichiers de données Parquet générés sont limités à 256 Mo et la taille des groupes de lignes est de 128 Mo. Le fichier manifeste se trouve au même niveau de dossier que les fichiers de données et présente le suffixe `manifest`.

```
unload ('select * from mytickit1.event')
to 's3://amzn-s3-demo-bucket/eventsize'
iam_role 'arn:aws:iam::123456789012:role/MyRedshiftRole'
maxfilesize 256 MB
rowgroupsize 128 MB
parallel on
parquet
manifest;
```

La liste de fichiers Amazon S3 se présente comme suit.

```
 Name                            Type      Last modified                         Size
 eventsize0000_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.5 KB
 eventsize0001_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.8 KB
 eventsize0002_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.4 KB
 eventsize0003_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.0 KB
 eventsize0004_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 25.3 KB
 eventsize0005_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 24.8 KB
 eventsize0006_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 25.0 KB
 eventsize0007_part_00.parquet parquet August 2, 2023, 17:35:21 (UTC-07:00) 25.6 KB
 eventsizemanifest                  -       August 2, 2023, 17:35:21 (UTC-07:00) 958.0 B
```

Le contenu du fichier `eventsizemanifest` se présente comme suit.

```
{
  "entries": [
    {"url":"s3://amzn-s3-demo-bucket/eventsize0000_part_00.parquet", "meta":
 { "content_length": 25130 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0001_part_00.parquet", "meta":
 { "content_length": 25428 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0002_part_00.parquet", "meta":
 { "content_length": 25025 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0003_part_00.parquet", "meta":
 { "content_length": 24554 }},
```

```
    {"url":"s3://amzn-s3-demo-bucket/eventsize0004_part_00.parquet", "meta":
 { "content_length": 25918 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0005_part_00.parquet", "meta":
 { "content_length": 25362 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0006_part_00.parquet", "meta":
 { "content_length": 25647 }},
    {"url":"s3://amzn-s3-demo-bucket/eventsize0007_part_00.parquet", "meta":
 { "content_length": 26256 }}
  ]
}
```

UPDATE

Rubriques

- Syntaxe
- Paramètres
- Notes d'utilisation
- Exemples d'instructions UPDATE

Met à jour les valeurs d'une ou de plusieurs Colonnes de la table lorsqu'une condition est satisfaite.

> **Note**
>
> La taille maximale d'une instruction SQL est de 16 Mo.

Syntaxe

```
[ WITH [RECURSIVE] common_table_expression [, common_table_expression , ...] ]
            UPDATE table_name [ [ AS ] alias ] SET column = { expression | DEFAULT }
 [,...]

[ FROM fromlist ]
[ WHERE condition ]
```

Paramètres

Clause WITH

Clause facultative qui en spécifie une ou plusieurs common-table-expressions. Consultez Clause WITH.

table_name

Table temporaire ou permanente. Seul le propriétaire de la table ou un utilisateur avec le privilège UPDATE sur la table peut-être mettre à jour les lignes. Si vous utilisez la clause FROM ou sélectionnez à partir de tables dans une expression ou une condition, vous devez avoir le privilège SELECT sur les tables. Vous ne pouvez pas ici donner un alias à la table ; cependant, vous pouvez spécifier un alias dans la clause FROM.

> **Note**
>
> Les tables externes d'Amazon Redshift Spectrum sont en lecture seule. Vous ne pouvez pas mettre à jour (UPDATE) une table externe.

alias

Nom alternatif temporaire d'une table cible. Les alias sont facultatifs. Le mot-clé AS est toujours facultatif.

SET colonne =

Une ou plusieurs colonnes que vous voulez modifier. Les colonnes qui ne sont pas répertoriées conservent leurs valeurs actuelles. N'incluez pas le nom de la table dans la spécification d'une colonne cible. Par exemple, `UPDATE tab SET tab.col = 1` n'est pas valide.

expression

Expression qui définit la nouvelle valeur de la colonne spécifiée.

DEFAULT

Met à jour la colonne avec la valeur par défaut qui a été attribuée à la colonne dans l'instruction CREATE TABLE.

FROM liste_tables

Vous pouvez mettre à jour une table en faisant référence aux informations d'autres tables. Répertoriez ces autres tables dans la clause FROM ou utilisez une sous-requête dans le cadre de la condition WHERE. Les tables répertoriées dans la clause FROM peuvent avoir des alias. Si vous avez besoin d'inclure la table cible de l'instruction UPDATE dans la liste, utilisez un alias.

WHERE condition

Clause facultative qui limite les mises à jour aux lignes qui correspondent à une condition. Lorsque la condition renvoie `true`, les colonnes SET spécifiées sont mises à jour. La condition peut être un simple prédicat sur une colonne ou une condition en fonction du résultat d'une sous-requête.

Vous pouvez nommer n'importe quelle table de la sous-requête, y compris la table cible de l'opération UPDATE.

Notes d'utilisation

Après la mise à jour d'un grand nombre de lignes dans une table :

- Exécuter une opération VACUUM sur la table pour récupérer de l'espace de stockage et retrier les lignes.
- Analysez la table pour mettre à jour les statistiques pour le planificateur de requête.

Les jointures externes complètes, gauche et droite ne sont pas prises en charge dans la clause FROM d'une instruction UPDATE ; elles renvoient l'erreur suivante :

```
ERROR: Target table must be part of an equijoin predicate
```

Si vous avez besoin de spécifier une jointure externe, utilisez une sous-requête dans la clause WHERE de l'instruction UPDATE.

Si votre instruction UPDATE nécessite une jointure réflexive avec la table cible, vous devez spécifier la condition de jointure, ainsi que les critères de la clause WHERE qui qualifient les lignes pour l'opération de mise à jour. En général, lorsque la table cible est associée à elle-même ou à une autre table, une bonne pratique consiste à utiliser une sous-requête qui sépare clairement les conditions de jointure des critères qui qualifient les lignes pour les mises à jour.

Les requêtes UPDATE avec plusieurs correspondances par ligne renvoient une erreur lorsque le paramètre de configuration `error_on_nondeterministic_update` est défini sur true. Pour plus d'informations, consultez error_on_nondeterministic_update.

Vous pouvez mettre à jour une colonne GENERATED BY DEFAULT AS IDENTITY. Les colonnes définies comme GENERATED BY DEFAULT AS IDENTITY peuvent être mises à jour avec des valeurs que vous fournissez. Pour plus d'informations, consultez GENERATED BY DEFAULT AS IDENTITY.

Exemples d'instructions UPDATE

Pour plus d'informations sur les tables utilisées dans les exemples suivants, consultez Exemple de base de données.

La table CATEGORY de la base de données TICKIT contient les lignes suivantes :

```
+-------+----------+-----------+--------------------------------------------+
| catid | catgroup |  catname  |                  catdesc                   |
+-------+----------+-----------+--------------------------------------------+
| 5     | Sports   | MLS       | Major League Soccer                        |
| 11    | Concerts | Classical | All symphony, concerto, and choir concerts |
| 1     | Sports   | MLB       | Major League Baseball                      |
| 6     | Shows    | Musicals  | Musical theatre                            |
| 3     | Sports   | NFL       | National Football League                   |
| 8     | Shows    | Opera     | All opera and light opera                  |
| 2     | Sports   | NHL       | National Hockey League                     |
| 9     | Concerts | Pop       | All rock and pop music concerts            |
| 4     | Sports   | NBA       | National Basketball Association            |
| 7     | Shows    | Plays     | All non-musical theatre                    |
| 10    | Concerts | Jazz      | All jazz singers and bands                 |
+-------+----------+-----------+--------------------------------------------+
```

Mise à jour d'une table en fonction d'une plage de valeurs

Mettez à jour la colonne CATGROUP en fonction d'une plage de valeurs de la colonne CATID.

```
UPDATE category
SET catgroup='Theatre'
WHERE catid BETWEEN 6 AND 8;

SELECT * FROM category
WHERE catid BETWEEN 6 AND 8;
```

```
+-------+----------+----------+--------------------------+
| catid | catgroup | catname  |          catdesc         |
+-------+----------+----------+--------------------------+
| 6     | Theatre  | Musicals | Musical theatre          |
| 7     | Theatre  | Plays    | All non-musical theatre  |
| 8     | Theatre  | Opera    | All opera and light opera |
+-------+----------+----------+--------------------------+
```

Mise à jour d'une table en fonction d'une valeur actuelle

Mettez à jour les colonnes CATNAME et CATDESC en fonction de leur valeur CATGROUP actuelle :

```
UPDATE category
SET catdesc=default, catname='Shows'
WHERE catgroup='Theatre';

SELECT * FROM category
WHERE catname='Shows';

+-------+----------+---------+---------+
| catid | catgroup | catname | catdesc |
+-------+----------+---------+---------+
| 6     | Theatre  | Shows   | NULL    |
| 7     | Theatre  | Shows   | NULL    |
| 8     | Theatre  | Shows   | NULL    |
+-------+----------+---------+---------+)
```

Dans ce cas, la colonne CATDESC a été définie sur null, car aucune valeur par défaut n'a été définie lors de la création de la table.

Exécutez les commandes suivantes pour redéfinir les données de la table CATEGORY à leurs valeurs d'origine :

```
TRUNCATE category;

COPY category
FROM 's3://redshift-downloads/tickit/category_pipe.txt'
DELIMITER '|'
IGNOREHEADER 1
REGION 'us-east-1'
IAM_ROLE default;
```

Mise à jour d'une table en fonction du résultat d'une sous-requête de clause WHERE

Mettez à jour la table CATEGORY en fonction du résultat d'une sous-requête de la clause WHERE :

```
UPDATE category
SET catdesc='Broadway Musical'
WHERE category.catid IN
(SELECT category.catid FROM category
JOIN event ON category.catid = event.catid
JOIN venue ON venue.venueid = event.venueid
JOIN sales ON sales.eventid = event.eventid
WHERE venuecity='New York City' AND catname='Musicals');
```

Affichez la table de mise à jour :

```
SELECT * FROM category ORDER BY catid;

+-------+----------+-----------+--------------------------------------------+
| catid | catgroup |  catname  |                  catdesc                   |
+-------+----------+-----------+--------------------------------------------+
| 2     | Sports   | NHL       | National Hockey League                     |
| 3     | Sports   | NFL       | National Football League                   |
| 4     | Sports   | NBA       | National Basketball Association            |
| 5     | Sports   | MLS       | Major League Soccer                        |
| 6     | Shows    | Musicals  | Broadway Musical                           |
| 7     | Shows    | Plays     | All non-musical theatre                    |
| 8     | Shows    | Opera     | All opera and light opera                  |
| 9     | Concerts | Pop       | All rock and pop music concerts            |
| 10    | Concerts | Jazz      | All jazz singers and bands                 |
| 11    | Concerts | Classical | All symphony, concerto, and choir concerts |
+-------+----------+-----------+--------------------------------------------+
```

Mise à jour d'une table en fonction du résultat d'une sous-requête de clause WITH

Pour mettre à jour la table CATEGORY en fonction du résultat d'une sous-requête à l'aide de la clause WITH, utilisez l'exemple suivant.

```
WITH u1 as (SELECT catid FROM event ORDER BY catid DESC LIMIT 1)
UPDATE category SET catid='200' FROM u1 WHERE u1.catid=category.catid;

SELECT * FROM category ORDER BY catid DESC LIMIT 1;

+-------+----------+---------+---------------------------------+
```

```
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
| 200   | Concerts | Pop     | All rock and pop music concerts |
+-------+----------+---------+---------------------------------+
```

Mise à jour d'une table en fonction du résultat d'une condition de jointure

Mettez à jour les 11 lignes originales de la table CATEGORY en fonction des lignes CATID correspondantes de la table EVENT :

```
UPDATE category SET catid=100
FROM event
WHERE event.catid=category.catid;

SELECT * FROM category ORDER BY catid;

+-------+----------+-----------+--------------------------------------------+
| catid | catgroup |  catname  |                 catdesc                    |
+-------+----------+-----------+--------------------------------------------+
| 2     | Sports   | NHL       | National Hockey League                     |
| 3     | Sports   | NFL       | National Football League                   |
| 4     | Sports   | NBA       | National Basketball Association            |
| 5     | Sports   | MLS       | Major League Soccer                        |
| 10    | Concerts | Jazz      | All jazz singers and bands                 |
| 11    | Concerts | Classical | All symphony, concerto, and choir concerts |
| 100   | Concerts | Pop       | All rock and pop music concerts            |
| 100   | Shows    | Plays     | All non-musical theatre                    |
| 100   | Shows    | Opera     | All opera and light opera                  |
| 100   | Shows    | Musicals  | Broadway Musical                           |
+-------+----------+-----------+--------------------------------------------+
```

Notez que la table EVENT apparaît dans la clause FROM et que la condition de jointure avec la table cible est définie dans la clause WHERE. Seules quatre lignes sont éligibles pour la mise à jour. Ces quatre lignes sont les lignes dont les valeurs CATID étaient à l'origine 6, 7, 8 et 9 ; seules ces quatre catégories sont représentées dans la table EVENT :

```
SELECT DISTINCT catid FROM event;

+-------+
| catid |
+-------+
| 6     |
```

```
| 7     |
| 8     |
| 9     |
+-------+
```

Mettez à jour les 11 lignes originales de la table CATEGORY en étendant l'exemple précédent et en ajoutant une autre condition à la clause WHERE. En raison de la restriction sur la colonne CATGROUP, une seule ligne est éligible à la mise à jour (même si quatre lignes sont qualifiées pour la jointure).

```
UPDATE category SET catid=100
FROM event
WHERE event.catid=category.catid
AND catgroup='Concerts';

SELECT * FROM category WHERE catid=100;

+-------+----------+---------+---------------------------------+
| catid | catgroup | catname |             catdesc             |
+-------+----------+---------+---------------------------------+
| 100   | Concerts | Pop     | All rock and pop music concerts |
+-------+----------+---------+---------------------------------+
```

Un autre moyen d'écrire cet exemple est le suivant :

```
UPDATE category SET catid=100
FROM event JOIN category cat ON event.catid=cat.catid
WHERE cat.catgroup='Concerts';
```

L'avantage de cette approche est que les critères de jointure sont clairement séparés des autres critères qui qualifient les lignes pour la mise à jour. Notez l'utilisation de l'alias CAT pour la table CATEGORY dans la clause FROM.

Mises à jour avec jointures externes dans la clause FROM

L'exemple précédent a montré une jointure interne spécifiée dans la clause FROM d'une instruction UPDATE. L'exemple suivant renvoie une erreur, car la clause FROM ne prend pas en charge les jointures externes avec la table cible :

```
UPDATE category SET catid=100
FROM event LEFT JOIN category cat ON event.catid=cat.catid
```

```
WHERE cat.catgroup='Concerts';
ERROR:  Target table must be part of an equijoin predicate
```

Si la jointure externe est nécessaire pour l'instruction UPDATE, vous pouvez déplacer la syntaxe de la jointure externe dans une sous-requête :

```
UPDATE category SET catid=100
FROM
(SELECT event.catid FROM event LEFT JOIN category cat ON event.catid=cat.catid)
 eventcat
WHERE category.catid=eventcat.catid
AND catgroup='Concerts';
```

Mises à jour avec les colonnes d'une autre table dans la clause SET

Pour mettre à jour la table listing de l'échantillon de base de données TICKIT avec les valeurs de la table sales, utilisez l'exemple suivant.

```
SELECT listid, numtickets FROM listing WHERE sellerid = 1 ORDER BY 1 ASC LIMIT 5;

+--------+------------+
| listid | numtickets |
+--------+------------+
| 100423 | 4          |
| 108334 | 24         |
| 117150 | 4          |
| 135915 | 20         |
| 205927 | 6          |
+--------+------------+

UPDATE listing
SET numtickets = sales.sellerid
FROM sales
WHERE sales.sellerid = 1 AND listing.sellerid = sales.sellerid;

SELECT listid, numtickets FROM listing WHERE sellerid = 1 ORDER BY 1 ASC LIMIT 5;

+--------+------------+
| listid | numtickets |
+--------+------------+
| 100423 | 1          |
| 108334 | 1          |
| 117150 | 1          |
```

```
| 135915 | 1          |
| 205927 | 1          |
+--------+------------+
```

USE

Modifie la base de données sur laquelle les requêtes sont exécutées. SHOW USE pointe vers la base de données la plus récemment utilisée avec la commande USE. RESET USE réinitialise la base de données utilisée. Cela signifie que si la base de données n'est pas spécifiée dans le SQL, les objets sont recherchés dans la base de données en cours.

Syntaxe

```
USE database
```

Exemples

Supposons qu'il existe trois bases de donnéespdb, `dev` et etpdb2. Supposons qu'il y ait deux tables `t` dans les schémas publics de chacune des bases de données. Tout d'abord, insérez les données dans les tables de différentes bases de données :

```
dev=# insert into dev.public.t values (1);
INSERT 0 1
dev=# insert into pdb.public.t values (2);
INSERT 0 1
```

Sans définir explicitement de base de données, le système utilise votre base de données connectée. Vérifiez le contexte actuel de votre base de données :

```
dev=# show use;
Use Database

(1 row)
dev=> show search_path;
search_path
$user, public
(1 row)
```

Lorsque vous interrogez une table `t` sans spécifier de base de données, le système utilise la table dans votre base de données actuelle :

```
dev=# select * from t;
c
----
1
(1 row)
```

Utilisez la `use` commande pour changer de base de données sans modifier votre connexion :

```
dev=# use pdb;
USE
dev=# show use;
 Use Database
--------------
 pdb
(1 row)
dev=# select * from t;
id
----
2
(1 row)
```

Vous pouvez également spécifier explicitement le schéma :

```
dev=# select * from public.t;
id
----
2
(1 row)
```

Vous pouvez désormais créer des tables dans différents schémas au sein de votre base de données actuelle :

```
dev=# create table s1.t(id int);
CREATE TABLE
dev=# insert into pdb.s1.t values (3);
INSERT 0 1
```

Le chemin de recherche détermine les objets du schéma auxquels vous accédez lorsque vous ne spécifiez aucun schéma :

```
dev=# set search_path to public, s1;
```

```
SET
dev=# select * from t;
 id
----
  2
(1 row)
```

Modifiez l'ordre des schémas pour accéder aux différentes tables :

```
dev=# set search_path to s1, public;
SET
dev=# show search_path;
 search_path
-------------
 s1, public
(1 row)
dev=# select * from t;
 id
----
  3
(1 row)
```

Basculez vers une autre base de données tout en conservant votre connexion d'origine :

```
dev=# show use;
 Use Database
--------------
 pdb
(1 row)
dev=# use pdb2;
USE
dev=# show use;
 Use Database
--------------
 pdb2
(1 row)
```

Lorsque vous changez de base de données, le chemin de recherche est réinitialisé par défaut :

```
dev=# show search_path;
  search_path
---------------
```

```
 $user, public
(1 row)
```

Créez une table et insérez des données dans votre base de données actuelle :

```
dev=# create table pdb2.public.t(id int);
CREATE TABLE
dev=# insert into pdb2.public.t values (4);
INSERT 0 1
dev=# select * from t;
 id
----
  4
(1 row)
```

Dans les transactions, vous pouvez écrire dans la base de données actuelle et lire dans n'importe quelle base de données à l'aide d'une notation en trois parties. Cela inclut également la base de données connectée :

```
dev=# show use;
 Use Database
--------------
 pdb2
(1 row)

dev=# BEGIN;
BEGIN
dev=# select * from t;
 id
----
  4
(1 row)

dev=# insert into t values (5);
INSERT 0 1
dev=# select * from t;
 id
----
  4
  5
(2 rows)
```

```
dev=# select * from pdb.public.t;
 id
----
  2
(1 row)

dev=# select * from dev.public.t;
 id
----
  1
(1 row)
```

Réinitialisez votre base de données connectée. Notez que cela revient non seulement à la base de données utilisée précédemmentpdb, mais également à la base de données connectée. Le chemin de recherche est également remplacé par celui par défaut :

```
dev=# RESET USE;
RESET
dev=# select * from t;
c
----
1
(1 row)
dev=# show use;
 Use Database
--------------

(1 row)

dev=# show search_path;
  search_path
---------------
 $user, public
(1 row)
```

Vous pouvez modifier les bases de données au début d'une transaction, mais pas après avoir exécuté des requêtes :

```
dev=# BEGIN;
BEGIN
dev=# use pdb;
USE
```

```
dev=# use pdb2;
USE
dev=# use pdb;
USE
dev=# select * from t;
 id
----
  2
(1 row)
dev=# use pdb2;
ERROR:  USEd Database cannot be set or reset inside a transaction after another
 command.
dev=# rollback;
ROLLBACK
(1 row)
```

Exemple de catalogue de données

Créez d'abord des tables dans différents schémas et catalogues pour illustrer les requêtes entre catalogues. Commencez par créer des tables dans la base de données connectée.

```
dev=# CREATE TABLE dev.public.t (col INT);
dev=# INSERT INTO dev.public.t VALUES (1);
dev=# CREATE SCHEMA write_schema;
dev=# CREATE TABLE dev.write_schema.t (state char (2));
dev=# INSERT INTO dev.write_schema.t VALUES ('WA');
```

Créez maintenant des tables similaires dans un autre catalogue. Cela montre comment utiliser des bases de données à catalogues multiples.

```
dev=# CREATE TABLE my_db@my_catalog.public.t (col INT);
dev=# INSERT INTO my_db@my_catalog.public.t VALUES (100);
dev=# CREATE SCHEMA my_db@my_catalog.write_schema;
dev=# CREATE TABLE my_db@my_catalog.write_schema.t (state char (2));
dev=# INSERT INTO my_db@my_catalog.write_schema.t VALUES ('CA');
```

Vérifiez le contexte actuel de la base de données. Sans définir explicitement de base de données, le système utilise la base de données connectée.

```
dev=# SHOW USE;
 Use Database
--------------

(1 row)

dev=# SHOW search_path;
  search_path
---------------
 $user, public
(1 row)

dev=# SELECT * FROM t;
 col
-----
   1
(1 row)
```

Configurez la USEd base de données pour qu'elle interroge les tables d'un autre catalogue.

```
dev=# USE my_db@my_catalog;

dev=# SHOW USE;
            Use Database
-------------------------------------
 my_db@my_catalog
(1 row)

dev=# SHOW search_path;
  search_path
---------------
 $user, public
(1 row)
```

Lorsque vous interrogez la table t, les résultats proviennent de la base de données inter-catalogues.

```
dev=# SELECT * FROM t;
 col
```

```
-----
 100
(1 row)

dev=# SELECT * FROM public.t;
 col
-----
 100
(1 row)

dev=# SELECT * FROM my_db@my_catalog.public.t;
 col
-----
 100
(1 row)
```

Modifiez le chemin de recherche pour accéder aux tables des différents schémas de la base de USEd données.

```
dev=# SET search_path to write_schema;

dev=# SHOW search_path;
 search_path
--------------
 write_schema
(1 row)

dev=# SELECT * FROM t;
 state
-------
 CA
(1 row)

dev=# SELECT * FROM write_schema.t;
 state
-------
 CA
(1 row)

dev=# SELECT * FROM my_db@my_catalog.write_schema.t;
 state
```

```
-------
 CA
(1 row)
```

Même si USE est défini sur une base de données inter-catalogues, il est toujours possible d'interroger explicitement la base de données d'origine.

```
dev=# SELECT * FROM dev.write_schema.t;
 state
-------
 WA
(1 row)
```

Réinitialisez la USEd base de données pour qu'elle fasse à nouveau référence aux objets de la base de données connectée.

```
dev=# RESET USE;

dev=# SHOW USE;
 Use Database
--------------

(1 row)
```

Notez que le search_path est réinitialisé lorsque USE est réinitialisé.

```
dev=# SHOW search_path;
  search_path
---------------
 $user, public
(1 row)
```

Après la réinitialisation, les requêtes font désormais référence à la base de données connectée d'origine.

```
dev=# SELECT * FROM t;
 col
-----
   1
(1 row)

dev=# SELECT * FROM public.t;
 col
-----
   1
(1 row)

dev=# SELECT * FROM dev.public.t;
 col
-----
   1
(1 row)
```

Vous pouvez modifier le chemin de recherche dans la base de données d'origine pour accéder à différents schémas.

```
dev=# SET search_path to write_schema;

dev=# SHOW search_path;
 search_path
--------------
 write_schema
(1 row)

dev=# SELECT * FROM t;
 state
-------
 WA
(1 row)

dev=# SELECT * FROM write_schema.t;
 state
-------
 WA
(1 row)
```

```
dev=# SELECT * FROM dev.write_schema.t;
 state
-------
 WA
(1 row)
```

VACUUM

Retrie les lignes et récupère l'espace dans une table spécifiée ou dans toutes les tables de la base de données actuelle.

> **Note**
>
> Seuls les utilisateurs disposant des autorisations nécessaires peuvent efficacement vider une table. Si VACUUM est exécutée sans les autorisations de table nécessaires, l'opération se termine correctement, mais n'a aucun effet. Pour obtenir la liste des autorisations de table valides permettant d'exécuter VACUUM efficacement, consultez Privilèges requis ci-après.

Amazon Redshift trie automatiquement les données et exécute l'opération VACUUM DELETE en arrière-plan. Cela réduit la nécessité d'exécuter la commande VACUUM. Pour plus d'informations, consultez Exécution de l'opération VACUUM sur les tables.

Par défaut, la commande VACUUM ignore la phase de tri pour toute table dans laquelle plus de 95 % des lignes sont déjà triées. L'omission de la phase de tri peut améliorer considérablement les performances de l'opération VACUUM. Pour modifier le seuil de suppression ou de tri par défaut d'une seule table, incluez le nom de la table et le paramètre TO seuil PERCENT lorsque vous exécutez la commande VACUUM.

Les utilisateurs peuvent accéder aux tables pendant qu'elles sont aspirées. Vous pouvez exécuter des requêtes et des opérations d'écriture pendant qu'une table est l'objet de la commande VACUUM, mais lorsque des commandes de langage de manipulation de données (DML) et une commande VACUUM s'exécutent en même temps, les deux peuvent prendre plus de temps. Si vous exécutez les instructions UPDATE et DELETE pendant une opération VACUUM, les performances du système peuvent être réduites. L'opération VACUUM DELETE bloque temporairement les opérations de mise à jour et de suppression.

Amazon Redshift exécute automatiquement une opération VACUUM DELETE ONLY en arrière-plan. L'opération automatique VACUUM s'interrompt lorsque les utilisateurs exécutent des opérations de langage de définition de données (DLL) telles que l'opération ALTER TABLE.

> **Note**
>
> La syntaxe et le comportement de la commande Amazon Redshift VACUUM sont sensiblement différentes de l'opération PostgreSQL VACUUM. Par exemple, l'opération VACUUM par défaut dans Amazon Redshift est VACUUM FULL, qui récupère de l'espace disque et retrie toutes les lignes. En revanche, l'opération VACUUM par défaut dans PostgreSQL récupère simplement l'espace et le rend disponible pour sa réutilisation.

Pour de plus amples informations, veuillez consulter Exécution de l'opération VACUUM sur les tables.

Privilèges requis

Les privilèges suivants sont requis pour VACUUM :

- Superuser
- Utilisateurs disposant du privilège VACUUM
- Propriétaire de la table
- Propriétaire de la base de données avec qui la table est partagée

Syntaxe

```
VACUUM [ FULL | SORT ONLY | DELETE ONLY | REINDEX | RECLUSTER ]
[ [ table_name ] [ TO threshold PERCENT ] [ BOOST ] ]
```

Paramètres

FULL

Trie la table spécifiée (ou toutes les tables de la base de données actuelle) et récupère l'espace disque occupé par les lignes qui ont été marquées en vue de leur suppression par les opérations UPDATE et DELETE précédentes. VACUUM FULL est la valeur par défaut.

Une opération complète n'effectue pas une réindexation des tables entrelacées. Pour réindexer les tables entrelacées et exécuter une opération VACUUM FULL, utilisez l'option VACUUM REINDEX.

Par défaut, VACUUM FULL ignore la phase de tri d'une table déjà triée à au moins 95 %. Si l'opération VACUUM peut ignorer la phase de tri, elle effectue DELETE ONLY et récupère l'espace de la phase de suppression de telle sorte qu'au moins 95 % des lignes restantes ne sont pas marquées en vue de leur suppression.

Si le seuil de tri n'est pas atteint (par exemple, si 90 % des lignes sont triées) et que VACUUM effectue un tri complet, il effectue également une opération de suppression complète, récupérant l'espace de 100 % des lignes supprimées.

Vous ne pouvez modifier le seuil par défaut de VACUUM que pour une seule table. Pour modifier le seuil par défaut de VACUUM pour une seule table, incluez le nom de la table et le paramètre TO seuil PERCENT.

SORT ONLY

Trie la table spécifiée (ou toutes les tables de la base de données actuelle) sans récupération de l'espace libéré par les lignes supprimées. Cette option est utile lorsque la récupération de l'espace disque n'est pas importante, mais qu'un nouveau tri l'est. Une opération SORT ONLY réduit le temps passé pour les opérations VACUUM lorsque la région non triée ne contient pas un grand nombre de lignes supprimées et ne couvre pas l'ensemble de la région triée. Les applications qui n'ont pas de contraintes d'espace disque, mais dépendent de l'optimisation des requêtes et de la conservation des lignes de table triées peuvent bénéficier de ce type d'opération.

Par défaut, VACUUM SORT ONLY ignore toute table déjà triée à au moins 95 %. Pour modifier le seuil de tri par défaut d'une seule table, incluez le nom de la table et le paramètre TO seuil PERCENT lorsque vous exécutez la commande VACUUM.

DELETE ONLY

Amazon Redshift exécute automatiquement une opération VACUUM DELETE ONLY en arrière-plan. Par conséquent, vous n'aurez que rarement, voire jamais, besoin d'exécuter une opération VACUUM DELETE ONLY.

L'opération VACUUM DELETE récupère l'espace disque occupé par les lignes marquées en vue de leur suppression par des opérations UPDATE et DELETE précédentes, et compacte la table pour libérer l'espace consommé. Une opération DELETE ONLY ne trie pas les données de la table.

Cette option réduit le temps passé pour les opérations VACUUM lorsque la récupération de l'espace disque est importante, mais qu'un nouveau tri des lignes ne l'est pas. Cette option peut également être utile lorsque vos performances de requête sont déjà optimales et qu'un nouveau tri des lignes pour optimiser ces performances n'est pas une condition requise.

Par défaut, VACUUM DELETE ONLY récupère de l'espace de telle sorte qu'au moins 95 % des lignes restantes ne sont pas marquées en vue de leur suppression. Pour modifier le seuil de suppression par défaut d'une seule table, incluez le nom de la table et le paramètre TO seuil PERCENT lorsque vous exécutez la commande VACUUM.

Certaines opérations, par exemple `ALTER TABLE APPEND`, peuvent provoquer la fragmentation des tables. Lorsque vous utilisez la clause `DELETE ONLY`, l'opération VACUUM récupère l'espace des tables fragmentées. La même valeur de seuil de 95% s'applique à l'opération de défragmentation.

REINDEX

Analyse la distribution des valeurs des colonnes de clé de tri entrelacé, puis effectue une opération VACUUM complète. Si REINDEX est utilisé, le nom d'une table est requis.

VACUUM REINDEX prend nettement plus de temps que VACUUM FULL, car l'opération effectue un passage supplémentaire pour analyser les clés de tri entrelacé. L'opération de tri et de fusion peut prendre plus de temps pour les tables entrelacées, car le tri entrelacé peut nécessiter la réorganisation de plus de lignes qu'un tri composé.

Si une opération VACUUM REINDEX se termine avant la fin, l'opération VACUUM suivante reprend l'opération de réindexation avant d'exécuter l'opération VACUUM complète.

VACUUM REINDEX n'est pas pris en charge avec TO seuil PERCENT.

REGROUPER

Trie les parties de la table qui ne sont pas triées. Les parties de la table qui sont déjà triées par le tri automatique de la table restent intactes. Cette commande ne fusionne pas les données nouvellement triées avec la région triée. Elle ne récupère pas non plus tout l'espace marqué pour suppression. Lorsque cette commande est terminée, il se peut que la table n'apparaisse pas entièrement triée, comme indiqué par le champ `unsorted` dans SVV_TABLE_INFO.

Nous vous recommandons d'utiliser VACUUM RECLUSTER pour les grandes tables avec une ingestion fréquente et des requêtes qui accèdent uniquement aux données les plus récentes.

VACUUM RECLUSTER n'est pas pris en charge avec TO threshold PERCENT. Si RECLUSTER est utilisé, le nom d'une table est requis.

VACUUM RECLUSTER n'est pas pris en charge sur les tables ayant des clés de tri entrelacées et les tables ayant le style de distribution ALL.

table_name

Nom de la table sur laquelle s'applique l'opération VACUUM. Si vous ne spécifiez pas un nom de table, l'opération VACUUM s'applique à toutes les tables de la base de données actuelle. Vous pouvez spécifier une table créée par l'utilisateur permanente ou temporaire. La commande n'est pas pertinente pour les autres objets, tels que les vues et les tables système.

Si vous incluez le paramètre TO seuil PERCENT, un nom de table est obligatoire.

TO seuil PERCENT

Clause qui spécifie le seuil au-dessus duquel VACUUM ignore la phase de tri et le seuil cible de récupération de l'espace dans la phase de suppression. Le seuil de tri est le pourcentage du total des lignes qui sont déjà dans l'ordre de tri de la table spécifiée avant l'opération VACUUM. Le seuil de suppression est le pourcentage minimal de lignes totales non marquées en vue de leur suppression après l'opération VACUUM.

Comme VACUUM ne retrie les lignes que lorsque le pourcentage de lignes triées d'une table est inférieur au seuil de tri, Amazon Redshift peut souvent réduire la durée de VACUUM de façon significative. De même, lorsque VACUUM n'est pas limité pour récupérer l'espace de 100 % des lignes marquée en vue de leur suppression, il est souvent possible d'ignorer la réécriture des blocs contenant seulement quelques lignes supprimées.

Par exemple, si vous spécifiez 75 pour seuil, VACUUM ignore la phase de tri si 75 % ou plus des lignes de la table sont déjà dans l'ordre de tri. Pour la phase de suppression, VACUUM définit une cible de récupération de l'espace disque de telle sorte qu'au moins 75 % des lignes de la table ne sont pas marquées en vue de leur suppression après l'opération VACUUM. La valeur seuil doit être un nombre entier compris entre 0 et 100. La valeur par défaut est 95. Si vous spécifiez une valeur égale à 100, VACUUM trie toujours la table, sauf si elle est déjà entièrement triée, et récupère l'espace de toutes les lignes marquées en vue de leur suppression. Si vous spécifiez la valeur 0, VACUUM ne trie jamais la table et ne récupère jamais l'espace.

Si vous incluez le paramètre TO seuil PERCENT, vous devez également spécifier un nom de table. Si le nom de table est omis, VACUUM échoue.

Vous ne pouvez pas utiliser le paramètre TO seuil PERCENT avec REINDEX.

BOOST

Exécute la commande VACUUM avec des ressources supplémentaires, comme la mémoire et l'espace disque, en fonction de leur disponibilité. Avec l'option BOOST, VACUUM exécute l'opération dans une fenêtre et bloque les suppressions et mises à jour simultanées pendant la durée de l'opération VACUUM. L'exécution de l'option BOOST utilise des ressources systèmes, ce qui peut affecter les performances de la requête. Exécutez l'opération VACUUM BOOST lorsque la charge sur le système est légère, par exemple pendant les opérations de maintenance.

Tenez compte des éléments suivants lorsque vous utilisez l'option BOOST :

- Lorsque l'option BOOST est spécifiée, la valeur table_name est requise.
- L'option BOOST n'est pas prise en charge avec REINDEX.
- L'option BOOST est ignorée avec DELETE ONLY.

Notes d'utilisation

Pour la plupart des applications Amazon Redshift, une opération VACUUM FULL est recommandée. Pour plus d'informations, consultez Exécution de l'opération VACUUM sur les tables.

Avant d'exécuter une opération VACUUM, notez le comportement suivant :

- Vous ne pouvez pas exécuter VACUUM au sein d'un bloc de transaction (BEGIN ... END). Pour plus d'informations sur les transactions, consultez Isolement sérialisable.
- Une certaine croissance de la table peut se produire lors d'une opération VACUUM. Ce comportement est attendu lorsqu'il n'y a aucune ligne supprimée à récupérer ou que le nouvel ordre de tri de la table se traduit par un ratio inférieur de compression des données.
- Pendant les opérations VACUUM, une certaine dégradation des performances des requêtes est attendue. Les performances normales reprennent dès que l'opération VACUUM est terminée.
- Les opérations d'écriture simultanées se poursuivent pendant les opérations VACUUM, mais nous ne les recommandons pas. Il est plus efficace de terminer les opérations d'écriture avant d'exécuter l'opération VACUUM. En outre, toutes les données écrites après le démarrage d'une opération de vide ne peuvent pas être nettoyées par cette opération. Dans ce cas, une deuxième opération de nettoyage est nécessaire.
- Une opération VACUUM peut ne pas être en mesure de démarrer si une opération de chargement ou d'insertion est déjà en cours. Les opérations VACUUM nécessitent temporairement un accès

exclusif aux tables afin de démarrer. Comme cet accès exclusif est requis brièvement, les opérations VACUUM ne bloquent pas les charges et les insertions simultanées pendant une période de temps significative.

- Les opérations VACUUM sont ignorées lorsqu'il n'y a aucun travail à faire pour une table spécifique ; cependant, la découverte que l'opération peut être ignorée a un coût. Si vous savez qu'une table est vierge ou qu'elle ne respecte pas le seuil VACUUM défini, n'exécutez pas sur elle d'opération VACUUM.
- Une opération DELETE ONLY sur une petite table peut ne pas réduire le nombre de blocs utilisés pour stocker les données, en particulier lorsque la table possède un grand nombre de colonnes ou que le cluster utilise un grand nombre de tranches par nœud. Ces opérations VACUUM ajoutent un bloc par colonne et par tranche pour tenir compte des insertions simultanées dans la table ; en outre, il est possible que cette surcharge compense la diminution du nombre de blocs de l'espace disque récupéré. Par exemple, si une table de 10 colonnes sur un cluster de 8 nœuds occupe 1000 blocs avant une opération VACUUM, celle-ci ne réduit pas le nombre réel de blocs, sauf si plus de 80 blocs d'espace disque sont récupérés en raison des lignes supprimées. (Chaque bloc de données utilise 1 Mo.)

Les opérations VACUUM automatiques s'interrompent si une des conditions suivantes est remplie :

- Un utilisateur exécute une opération de langage de manipulation de données (DDL), telle que ALTER TABLE, qui nécessite un verrou exclusif sur une table pour laquelle une opération VACUUM automatique est en cours.
- Une période de forte charge de cluster.

Support pour VACUUM simultané

Amazon Redshift prend en charge l'exécution simultanée de plusieurs transactions sous vide sur différentes sessions d'un cluster ou d'un groupe de travail. Cela signifie que vous pouvez émettre des instances différentes et multiples de tous les modes d'aspiration à la fois, chaque transaction d'aspiration étant enregistrée sur une table unique. Deux opérations d'aspiration ne peuvent pas fonctionner sur une même table en même temps.

Directives relatives à l'utilisation simultanée d'un aspirateur

- Lorsque vous exécutez des transactions de vide simultanées sur différentes sessions, vous devez surveiller les ressources du système et éviter d'exécuter un trop grand nombre d'opérations de vide simultanément.

- Le niveau de simultanéité recommandé dépend de l'espace à récupérer, du nombre et de la largeur des rangées à trier, de la taille de l'entrepôt et de la taille de votre charge de travail associée aux opérations VACUUM.
- Selon le mode de transaction de mise sous vide, commencez par deux opérations de mise sous vide simultanées, puis ajoutez-en d'autres en fonction de leur durée d'exécution et de la charge du système. Tout comme les autres requêtes volumineuses émises par les utilisateurs, les opérations de vide peuvent commencer à être mises en file d'attente si vous en exécutez un trop grand nombre simultanément lorsqu'Amazon Redshift atteint les limites de ressources du système.
- Exécutez plusieurs opérations Vacuum BOOST avec soin. Exécuter Vacuum avec l'option BOOST sollicite les ressources du système, ce qui peut affecter les performances des requêtes. Exécutez l’opération VACUUM BOOST lorsque la charge sur le système est légère, par exemple pendant les opérations de maintenance.
- Si vous ne spécifiez pas un nom de table, l’opération VACUUM s’applique à toutes les tables de la base de données actuelle. Ces opérations de mise sous vide se déroulent toujours de manière séquentielle.

Exemples

Récupérez l’espace et la base de données, et triez à nouveau les lignes de toutes les tables en fonction du seuil d’aspiration de 95 % par défaut.

```
vacuum;
```

Récupérez l’espace et retriez les lignes de la table SALES en fonction du seuil de 95 % par défaut.

```
vacuum sales;
```

Récupérez toujours l’espace et retriez les lignes de la table SALES.

```
vacuum sales to 100 percent;
```

Retriez les lignes de la table SALES uniquement si moins de 75 % des lignes sont déjà triées.

```
 vacuum sort only sales to 75 percent;
```

Récupérez de l’espace dans la table SALES, de telle sorte qu’au moins 75 % des lignes restantes ne soient pas marqués en vue de leur suppression après l’aspiration.

```
vacuum delete only sales to 75 percent;
```

Réindexez, puis aspirez la table LISTING.

```
vacuum reindex listing;
```

La commande suivante renvoie une erreur.

```
vacuum reindex listing to 75 percent;
```

Regroupez, puis aspirez la table LISTING.

```
vacuum recluster listing;
```

Regroupez, puis aspirez la table LISTING avec l'option BOOST.

```
vacuum recluster listing boost;
```

Référence sur les fonctions SQL

Rubriques

- Fonctions exécutées uniquement sur le nœud principal
- Fonctions d'agrégation
- Fonctions de tableau
- Fonctions d'agrégation bit par bit
- Expressions conditionnelles
- Fonctions de formatage des types de données
- Fonctions de date et d'heure
- Fonctions de hachage
- HyperLogLog fonctions
- Fonctions JSON
- Solutions de machine learning

- Fonctions mathématiques
- Fonctions d'objet
- Fonctions spatiales
- Fonctions de chaîne
- Fonctions d'informations sur le type SUPER
- Fonctions et opérateurs VARBYTE
- Fonctions de fenêtrage
- Fonctions d'administration système
- Fonctions d'informations système

Amazon Redshift prend en charge un certain nombre de fonctions qui sont des extensions de la norme SQL, ainsi que des fonctions d'agrégation standard, des fonctions scalaires et des fonctions de fenêtrage.

> **Note**
>
> Amazon Redshift est basé sur PostgreSQL. Amazon Redshift et PostgreSQL présentent un certain nombre de différences très importantes dont vous devez être conscient lorsque vous concevez et développez vos applications d'entrepôt des données. Pour plus d'informations sur les différences entre Amazon Redshift SQL et PostgreSQL, consultez Amazon Redshift et PostgreSQL.

Fonctions exécutées uniquement sur le nœud principal

Certaines requêtes Amazon Redshift sont distribuées et exécutées sur les nœuds de calcul, et d'autres requêtes s'exécutent exclusivement sur le nœud principal.

Le nœud principal répartit SQL sur les nœuds de calcul chaque fois qu'une requête fait référence aux tables créées par l'utilisateur ou aux tables système (tables avec le préfixe STL ou STV, et vues système avec le préfixe SVL ou SVV). Une requête qui ne fait référence qu'aux tables catalogue (tables avec le préfixe PG, comme PG_TABLE_DEF) ou qui ne fait référence à aucune table, s'exécute exclusivement sur le nœud principal.

Certaines fonctions SQL d'Amazon Redshift sont prises en charge uniquement sur le nœud principal et ne le sont pas sur les nœuds de calcul. Une requête qui utilise une fonction de nœud principal

doit être exécutée exclusivement sur celui-ci, et non sur les nœuds de calcul, sinon elle renverra une erreur.

La documentation relative à chaque fonction de nœud principal uniquement inclut un commentaire attestant que la fonction renvoie une erreur si elle fait référence aux tables définies par l'utilisateur ou aux tables système Amazon Redshift.

Pour plus d'informations, consultez Fonctions SQL prises en charge sur le nœud principal.

Les fonctions SQL suivantes sont des fonctions de nœud principal uniquement et ne sont pas prises en charge sur les nœuds de calcul :

Fonctions d'informations système

- CURRENT_SCHEMA
- CURRENT_SCHEMAS
- HAS_DATABASE_PRIVILEGE
- HAS_SCHEMA_PRIVILEGE
- HAS_TABLE_PRIVILEGE

Fonctions de chaîne

- SUBSTR

Fonctions mathématiques

- FACTORIELLE ()

Les fonctions de nœud principal uniquement suivantes sont déconseillées et ne sont plus prises en charge :

Fonctions de date

- AGE
- CURRENT_TIME
- CURRENT_TIMESTAMP
- LOCALTIME

- ISFINITE
- NOW

Fonctions de chaîne

- GETBIT
- GET_BYTE
- SET_BIT
- SET_BYTE
- TO_ASCII

Fonctions d’agrégation

Rubriques

- Fonction ANY_VALUE
- Fonction APPROXIMATE PERCENTILE_DISC
- Fonction AVG
- Fonction COUNT
- Fonction LISTAGG
- Fonction MAX
- Fonction MEDIAN
- Fonction MIN
- Fonction PERCENTILE_CONT
- Fonctions STDDEV_SAMP et STDDEV_POP
- Fonction SUM
- Fonctions VAR_SAMP et VAR_POP

Les fonctions d’agrégation calculent une valeur de résultat unique à partir d’un ensemble de valeurs d’entrée.

Les instructions SELECT utilisant des fonctions d’agrégation peuvent inclure deux clauses facultatives : GROUP BY et HAVING. La syntaxe de ces clauses est la suivante (avec la fonction COUNT par exemple) :

```
SELECT count (*) expression FROM table_reference
WHERE condition [GROUP BY expression ] [ HAVING condition]
```

La clause GROUP BY agrège et regroupe les résultats en fonction de valeurs uniques dans une ou des colonnes spécifiées. La clause HAVING limite les résultats renvoyés aux lignes dans lesquelles une condition d'agrégation particulière a la valeur true, par exemple count (*) > 1. La clause HAVING est utilisée de la même manière que WHERE afin de limiter les lignes en fonction de la valeur d'une colonne. Pour voir un exemple de ces clauses supplémentaires, consultez COUNT.

Les fonctions d'agrégation n'acceptent pas les fonctions d'agrégation ou les fonctions de fenêtrage imbriquées comme arguments.

Fonction ANY_VALUE

La fonction ANY_VALUE renvoie n'importe quelle valeur des valeurs d'expression en entrée de manière non déterministe. Cette fonction renvoie la valeur NULL si l'expression en entrée n'entraîne pas de renvoi de ligne. La fonction peut également renvoyer NULL si l'expression d'entrée contient des valeurs NULL.

Syntaxe

```
ANY_VALUE( [ DISTINCT | ALL ] expression )
```

Arguments

DISTINCT | ALL

Spécifiez DISTINCT ou ALL pour renvoyer n'importe quelle valeur des valeurs d'expression en entrée. L'argument DISTINCT n'a aucun effet et est ignoré.

expression

Colonne cible ou expression sur laquelle la fonction opère. L'expression est l'un des types de données suivants :

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL

- REAL
- DOUBLE PRECISON
- BOOLEAN
- CHAR
- VARCHAR
- DATE
- TIMESTAMP
- TIMESTAMPTZ
- TIME
- TIMETZ
- INTERVALLE D'UNE ANNÉE À L'AUTRE
- INTERVALLE D'UN JOUR À L'AUTRE
- VARBYTE
- SUPER
- HLLSKETCH
- GEOMETRY
- GEOGRAPHY

Renvoie

Renvoie le même type de données que expression.

Notes d’utilisation

Si une instruction qui spécifie la fonction ANY_VALUE d’une colonne inclut également une deuxième référence de colonne, la deuxième colonne doit apparaître dans une clause GROUP BY ou être incluse dans une fonction d’agrégation.

Exemples

Les exemples utilisent la table d’événements créée à l’étape 4 : charger des exemples de données depuis Amazon S3 du Guide de démarrage d’Amazon Redshift. L’exemple suivant renvoie une instance de n’importe quel dateid dont le nom d’événement est Eagles.

```
select any_value(dateid) as dateid, eventname from event where eventname ='Eagles'
 group by eventname;
```

Voici les résultats.

```
dateid | eventname
-------+--------------
 1878  | Eagles
```

L'exemple suivant renvoie une instance de n'importe quel dateid dont le nom d'événement est Eagles ou Cold War Kids.

```
select any_value(dateid) as dateid, eventname from event where eventname in('Eagles',
 'Cold War Kids') group by eventname;
```

Voici les résultats.

```
dateid | eventname
-------+--------------
 1922  | Cold War Kids
 1878  | Eagles
```

Fonction APPROXIMATE PERCENTILE_DISC

APPROXIMATE PERCENTILE_DISC est une fonction de distribution inverse qui suppose un modèle de distribution discrète. Elle prend une valeur de centile et une spécification de tri et renvoie un élément de l'ensemble donné. L'approximation permet une exécution de la fonction nettement plus rapide, avec une erreur relative faible d'environ 0,5 %.

Pour une valeur de percentile donnée, APPROXIMATE PERCENTILE_DISC utilise un algorithme résumé de quantile afin d'évaluer de façon approximative le percentile discret de l'expression dans la clause ORDER BY. APPROXIMATE PERCENTILE_DISC renvoie la valeur ayant la valeur de distribution cumulative la moins élevée (par rapport à la même spécification de tri) supérieure ou égale au percentile.

Syntaxe

```
APPROXIMATE  PERCENTILE_DISC ( percentile )
WITHIN GROUP (ORDER BY expr)
```

Arguments

percentile

Constante numérique comprise entre 0 et 1. Les valeurs NULL sont ignorées dans le calcul.

WITHIN GROUP (ORDER BY expr)

Clause qui spécifie les date/time valeurs numériques ou sur lesquelles trier et calculer le percentile.

Renvoie

Type de données identique à l'expression ORDER BY dans la clause WITHIN GROUP.

Notes d'utilisation

Si l'instruction APPROXIMATE PERCENTILE_DISC inclut une clause GROUP BY, le jeu de résultats est limité. La limite varie en fonction du type de nœud et du nombre de nœuds. Si la limite est dépassée, la fonction échoue et renvoie l'erreur suivante.

```
GROUP BY limit for approximate percentile_disc exceeded.
```

Si vous devez évaluer plus de groupes que ne le permet la limite, pensez à utiliser Fonction PERCENTILE_CONT.

Exemples

L'exemple suivant renvoie le nombre de ventes, le total des ventes et la valeur du 50e percentile pour les 10 meilleures dates.

```
select top 10 date.caldate,
count(totalprice), sum(totalprice),
approximate percentile_disc(0.5)
within group (order by totalprice)
from listing
join date on listing.dateid = date.dateid
group by date.caldate
order by 3 desc;

caldate    | count | sum        | percentile_disc
-----------+-------+------------+----------------
2008-01-07 |   658 | 2081400.00 |         2020.00
```

```
2008-01-02 |   614 | 2064840.00 |         2178.00
2008-07-22 |   593 | 1994256.00 |         2214.00
2008-01-26 |   595 | 1993188.00 |         2272.00
2008-02-24 |   655 | 1975345.00 |         2070.00
2008-02-04 |   616 | 1972491.00 |         1995.00
2008-02-14 |   628 | 1971759.00 |         2184.00
2008-09-01 |   600 | 1944976.00 |         2100.00
2008-07-29 |   597 | 1944488.00 |         2106.00
2008-07-23 |   592 | 1943265.00 |         1974.00
```

Fonction AVG

La fonction AVG renvoie la moyenne (arithmétique) des valeurs d'expression d'entrée. La fonction AVG utilise des valeurs numériques et ignore les valeurs NULL.

Syntaxe

```
AVG ( [ DISTINCT | ALL ] expression )
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère. L'expression est l'un des types de données suivants :

- SMALLINT
- INTEGER
- BIGINT
- NUMERIC
- DECIMAL
- REAL
- DOUBLE PRECISON
- SUPER

DISTINCT | ALL

Avec l'argument DISTINCT, la fonction supprime toutes les valeurs en double dans l'expression spécifiée avant de calculer la moyenne. Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression pour calculer la moyenne. La valeur par défaut est ALL.

Types de données

Les types d'argument pris en charge par la fonction AVG sont SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL, DOUBLE PRECISION et SUPER.

Les types de retour pris en charge par la fonction AVG sont les suivants :

- BIGINT pour tout argument de type nombre entier
- DOUBLE PRECISION pour un argument à virgule flottante
- Renvoie le même type de données que l'expression pour tout autre type d'argument.

La précision par défaut d'un résultat de fonction AVG avec un argument NUMERIC ou DECIMAL est de 38. L'échelle du résultat est identique à celle de l'argument. Par exemple, une fonction AVG d'une colonne DEC(5,2) renvoie un type de données DEC(38,2).

Exemples

Recherchez la quantité moyenne vendue par transaction dans la table SALES :

```
select avg(qtysold)from sales;

avg
-----
2
(1 row)
```

Recherchez le prix total moyen répertorié dans toutes les listes :

```
select avg(numtickets*priceperticket) as avg_total_price from listing;

avg_total_price
-----------------
3034.41
(1 row)
```

Recherchez le prix moyen payé, regroupé par mois par ordre décroissant :

```
select avg(pricepaid) as avg_price, month
from sales, date
```

```
where sales.dateid = date.dateid
group by month
order by avg_price desc;

avg_price | month
----------+-------
659.34 | MAR
655.06 | APR
645.82 | JAN
643.10 | MAY
642.72 | JUN
642.37 | SEP
640.72 | OCT
640.57 | DEC
635.34 | JUL
635.24 | FEB
634.24 | NOV
632.78 | AUG
(12 rows)
```

Fonction COUNT

La fonction COUNT compte les lignes définies par l'expression.

La fonction COUNT présente les variantes suivantes.

- COUNT (*) compte toutes les lignes de la table cible, qu'elles comprennent des valeurs null ou non.
- COUNT (expression) calcule le nombre de lignes avec des valeurs non NULL dans une colonne ou une expression spécifique.
- COUNT (DISTINCT expression) calcule le nombre de valeurs non NULL distinctes dans une colonne ou une expression.
- APPROXIMATE COUNT DISTINCT donne une approximation du nombre de valeurs distinctes non NULL dans une colonne ou une expression.

Syntaxe

```
COUNT( * | expression )
```

```
COUNT ( [ DISTINCT | ALL ] expression )
```

```
APPROXIMATE COUNT ( DISTINCT expression )
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère. La fonction COUNT prend en charge tous les types de données d'argument.

DISTINCT | ALL

Avec l'argument DISTINCT, la fonction supprime toutes les valeurs en double dans l'expression spécifiée avant d'effectuer le compte. Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression pour le compte. La valeur par défaut est ALL.

APPROXIMATE

Lorsqu'elle est utilisée avec APPROXIMATE, une fonction COUNT DISTINCT utilise un HyperLogLog algorithme pour estimer le nombre de valeurs distinctes non NULL dans une colonne ou une expression. Les requêtes qui utilisent le mot clé APPROXIMATE s'exécutent beaucoup plus rapidement, avec une erreur relative faible d'environ 2 %. L'approximation est garantie pour les requêtes qui renvoient un grand nombre de valeurs distinctes (par millions ou plus encore) par requête, ou par groupe, en cas de clause GROUP BY. Pour les ensembles plus petits de valeurs distinctes (par milliers), l'approximation peut être plus lente qu'un compte précis. La fonction APPROXIMATE peut être utilisée uniquement avec COUNT DISTINCT.

Type de retour

La fonction COUNT renvoie BIGINT.

Exemples

Pour compter tous les utilisateurs de l'état de Floride :

```
select count(*) from users where state='FL';

count
-------
510
```

Comptez tous les noms d'événements de la table EVENT :

```
select count(eventname) from event;

count
-------
8798
```

Comptez tous les noms d'événements de la table EVENT :

```
select count(all eventname) from event;

count
-------
8798
```

Comptez tous les lieux uniques dans le tableau IDs des ÉVÉNEMENTS :

```
select count(distinct venueid) as venues from event;

venues
--------
204
```

Pour compter le nombre de fois où chaque vendeur a répertorié des lots de plus de quatre billets en vente. Pour regrouper les résultats de l'ID du vendeur :

```
select count(*), sellerid from listing
where numtickets > 4
group by sellerid
order by 1 desc, 2;

count | sellerid
------+----------
12    |    6386
11    |    17304
11    |    20123
11    |    25428
...
```

Les exemples suivants comparent les valeurs de retour et les durées d'exécution de COUNT et de APPROXIMATE COUNT.

```
select  count(distinct pricepaid) from sales;

count
-------
  4528


Time: 48.048 ms


select approximate count(distinct pricepaid) from sales;

count
-------
  4553


Time: 21.728 ms
```

Fonction LISTAGG

Pour chaque groupe d'une requête, la fonction d'agrégation LISTAGG trie les lignes du groupe conformément à l'expression ORDER BY, puis concatène les valeurs en une chaîne unique.

Syntaxe

```
LISTAGG( [DISTINCT] aggregate_expression [, 'delimiter' ] )
[ WITHIN GROUP (ORDER BY order_list) ]
```

Arguments

DISTINCT

Clause qui supprime toutes les valeurs en double dans l'expression spécifiée avant de procéder à la concaténation. Les espaces de fin sont ignorés. Par exemple, les chaînes `'a'` et `'a  '` sont traitées comme des doublons. LISTAGG utilise la première valeur rencontrée. Pour plus d'informations, consultez Signification des blancs de fin.

aggregate_expression

Toute expression valide, comme un nom de colonne, qui fournit les valeurs à regrouper. Les valeurs NULL et les chaînes vides sont ignorées.

delimiter

Constante de chaîne qui sépare les valeurs concaténées. La valeur par défaut est NULL.

WITHIN GROUP (ORDER BY order_list)

Clause qui spécifie l'ordre de tri des valeurs regroupées.

Renvoie

VARCHAR(MAX). Si le jeu de résultats est supérieur à la taille de VARCHAR maximale, LISTAGG renvoie l'erreur suivante :

```
Invalid operation: Result size exceeds LISTAGG limit
```

Notes d'utilisation

- Si une instruction inclut plusieurs fonctions LISTAGG qui utilisent des clauses WITHIN GROUP, chaque clause WITHIN GROUP doit utiliser les mêmes valeurs ORDER BY.

  Par exemple, l'instruction suivante renvoie une erreur.

  ```
  SELECT LISTAGG(sellerid)
  WITHIN GROUP (ORDER BY dateid) AS sellers,
  LISTAGG(dateid)
  WITHIN GROUP (ORDER BY sellerid) AS dates
  FROM sales;
  ```

  L'instruction suivante s'exécute avec succès.

  ```
  SELECT LISTAGG(sellerid)
  WITHIN GROUP (ORDER BY dateid) AS sellers,
  LISTAGG(dateid)
  WITHIN GROUP (ORDER BY dateid) AS dates
  FROM sales;

  SELECT LISTAGG(sellerid)
  WITHIN GROUP (ORDER BY dateid) AS sellers,
  LISTAGG(dateid) AS dates
  FROM sales;
  ```

- Vous ne pouvez pas utiliser les fonctions d'agrégation LISTAGG, PERCENTILE_CONT et MEDIAN avec d'autres fonctions d'agrégation distinctes.

Exemples

L'exemple suivant regroupe les vendeurs IDs, triés par numéro de vendeur.

```
SELECT LISTAGG(sellerid, ', ')
WITHIN GROUP (ORDER BY sellerid)
FROM sales
WHERE eventid = 4337;

listagg

-------------------------------------------------------------------------------------------------
380, 380, 1178, 1178, 1178, 2731, 8117, 12905, 32043, 32043, 32043, 32432, 32432,
 38669, 38750, 41498, 45676, 46324, 47188, 47188, 48294
```

L'exemple suivant utilise DISTINCT pour renvoyer une liste de vendeurs uniques IDs.

```
SELECT LISTAGG(DISTINCT sellerid, ', ')
WITHIN GROUP (ORDER BY sellerid)
FROM sales
WHERE eventid = 4337;

listagg

-----------------------------------------------------------------------------------------
380, 1178, 2731, 8117, 12905, 32043, 32432, 38669, 38750, 41498, 45676, 46324, 47188,
 48294
```

L'exemple suivant regroupe les vendeurs par ordre IDs chronologique.

```
SELECT LISTAGG(sellerid, ', ')
WITHIN GROUP (ORDER BY dateid)
FROM sales
WHERE eventid = 4337;

   listagg
-------------------------------------------------------------------------------------------------
```

```
 41498, 47188, 47188, 1178, 1178, 1178, 380, 45676, 46324, 48294, 32043, 32043, 32432,
 12905, 8117, 38750, 2731, 32432, 32043, 380, 38669
```

L'exemple suivant renvoie une liste des dates de ventes de l'acheteur ID 660 séparées par des barres verticales.

```
SELECT LISTAGG(
    (SELECT caldate FROM date WHERE date.dateid=sales.dateid), ' | '
)
WITHIN GROUP (ORDER BY sellerid DESC, salesid ASC)
FROM sales
WHERE buyerid = 660;

             listagg
-------------------------------------------------
2008-07-16 | 2008-07-09 | 2008-01-01 | 2008-10-26
```

L'exemple suivant renvoie une liste des ventes séparées par des IDs virgules pour les acheteurs IDs 660, 661 et 662.

```
SELECT buyerid,
LISTAGG(salesid,', ')
WITHIN GROUP (ORDER BY salesid) AS sales_id
FROM sales
WHERE buyerid BETWEEN 660 AND 662
GROUP BY buyerid
ORDER BY buyerid;

buyerid |                sales_id
--------+-------------------------------------------------------
660     | 32872, 33095, 33514, 34548
661     | 19951, 20517, 21695, 21931
662     | 3318, 3823, 4215, 51980, 53202, 55908, 57832, 171603
```

Fonction MAX

La fonction MAX renvoie la valeur maximale d'un ensemble de lignes. La fonction DISTINCT ou ALL peut être utilisée, mais elle n'affecte pas le résultat.

Syntaxe

```
MAX ( [ DISTINCT | ALL ] expression )
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère. L'expression est l'un des types de données suivants :

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISON
- CHAR
- VARCHAR
- DATE
- TIMESTAMP
- TIMESTAMPTZ
- TIME
- TIMETZ
- VARBYTE
- SUPER

DISTINCT | ALL

Avec l'argument DISTINCT, la fonction supprime toutes les valeurs en double dans l'expression spécifiée avant de calculer la valeur maximale. Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression pour calculer la valeur maximale. La valeur par défaut est ALL.

Types de données

Renvoie le même type de données que expression. L'équivalent booléen de la fonction MIN est Fonction BOOL_AND, et l'équivalent booléen de MAX est Fonction BOOL_OR.

Exemples

Recherchez le prix le plus élevé payé de toutes les ventes :

```
select max(pricepaid) from sales;

max
----------
12624.00
(1 row)
```

Pour trouver le prix le plus élevé payé par billet de toutes les ventes :

```
select max(pricepaid/qtysold) as max_ticket_price
from sales;

max_ticket_price
-----------------
2500.00000000
(1 row)
```

Fonction MEDIAN

Calcule la valeur médiane de la plage de valeurs. Les valeurs NULL de la plage sont ignorées.

MEDIAN est une fonction de distribution inverse qui suppose un modèle de distribution continue.

MEDIAN est un cas particulier de PERCENTILE_CONT.

Syntaxe

```
MEDIAN(median_expression)
```

Arguments

median_expression

Colonne cible ou expression sur laquelle la fonction opère.

Types de données

Le type de retour est déterminé par le type de données de median_expression. Le tableau suivant illustre le type de retour de chaque type de données median_expression.

Type d'entrée	Type de retour
INT2, INT4, INT8, NUMERIC, DECIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATE	DATE
TIMESTAMP	TIMESTAMP
TIMESTAMPTZ	TIMESTAMPTZ

Notes d'utilisation

Si l'argument median_expression est un type de données DECIMAL défini avec la précision maximale de 38 chiffres, il est possible que MEDIAN renvoie un résultat inexact ou une erreur. Si la valeur de retour de la fonction MEDIAN dépasse 38 chiffres, le résultat est tronqué pour s'adapter, ce qui entraîne une perte de précision. Si, au cours de l'interpolation, un résultat intermédiaire dépasse la précision maximale, un dépassement de capacité numérique se produit et la fonction renvoie une erreur. Pour éviter ces conditions, nous vous recommandons d'utiliser un type de données avec une précision inférieure ou l'argument median_expression avec une précision inférieure.

Si une instruction inclut plusieurs appels à des fonctions d'agrégation basées sur le tri (LISTAGG, PERCENTILE_CONT, or MEDIAN), elles doivent toutes utiliser les mêmes valeurs ORDER BY. Notez que MEDIAN applique une clause order by implicite sur la valeur d'expression.

Par exemple, l'instruction suivante renvoie une erreur.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

An error occurred when executing the SQL command:
```

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

ERROR: within group ORDER BY clauses for aggregate functions must be the same
```

L'instruction suivante s'exécute avec succès.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(salesid)
FROM sales
GROUP BY salesid, pricepaid;
```

Exemples

Les exemples suivants utilisent l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

L'exemple suivant montre que MEDIAN produit les mêmes résultats que PERCENTILE_CONT(0.5).

```
SELECT TOP 10 DISTINCT sellerid, qtysold,
PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY qtysold),
MEDIAN(qtysold)
FROM sales
GROUP BY sellerid, qtysold;

+----------+---------+-----------------+--------+
| sellerid | qtysold | percentile_cont | median |
+----------+---------+-----------------+--------+
|        2 |       2 |               2 |      2 |
|       26 |       1 |               1 |      1 |
|       33 |       1 |               1 |      1 |
|       38 |       1 |               1 |      1 |
|       43 |       1 |               1 |      1 |
|       48 |       2 |               2 |      2 |
|       48 |       3 |               3 |      3 |
|       77 |       4 |               4 |      4 |
|       85 |       4 |               4 |      4 |
|       95 |       2 |               2 |      2 |
```

```
+----------+---------+-----------------+--------+
```

L'exemple suivant permet de trouver la quantité médiane vendue pour chaque ID de vendeur.

```
SELECT sellerid,
MEDIAN(qtysold)
FROM sales
GROUP BY sellerid
ORDER BY sellerid
LIMIT 10;

+----------+--------+
| sellerid | median |
+----------+--------+
|        1 |    1.5 |
|        2 |      2 |
|        3 |      2 |
|        4 |      2 |
|        5 |      1 |
|        6 |      1 |
|        7 |    1.5 |
|        8 |      1 |
|        9 |      4 |
|       12 |      2 |
+----------+--------+
```

Pour vérifier les résultats de la requête précédente pour le premier ID de vendeur, utilisez l'exemple suivant.

```
SELECT qtysold
FROM sales
WHERE sellerid=1;

+---------+
| qtysold |
+---------+
|       2 |
|       1 |
+---------+
```

Fonction MIN

La fonction MIN renvoie la valeur minimale d'un ensemble de lignes. La fonction DISTINCT ou ALL peut être utilisée, mais elle n'affecte pas le résultat.

Syntaxe

```
MIN ( [ DISTINCT | ALL ] expression )
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère. L'expression est l'un des types de données suivants :

- SMALLINT
- INTEGER
- BIGINT
- DECIMAL
- REAL
- DOUBLE PRECISON
- CHAR
- VARCHAR
- DATE
- TIMESTAMP
- TIMESTAMPTZ
- TIME
- TIMETZ
- VARBYTE
- SUPER

DISTINCT | ALL

Avec l'argument DISTINCT, la fonction supprime toutes les valeurs en double dans l'expression spécifiée avant de calculer la valeur minimale. Avec l'argument ALL, la fonction conserve toutes

les valeurs en double de l'expression pour calculer la valeur minimale. La valeur par défaut est ALL.

Types de données

Renvoie le même type de données que expression. L'équivalent booléen de la fonction MIN est Fonction BOOL_AND, et l'équivalent booléen de MAX est Fonction BOOL_OR.

Exemples

Pour trouver le prix le plus bas payé de toutes les ventes :

```
select min(pricepaid) from sales;

min
-------
20.00
(1 row)
```

Pour trouver le prix le plus bas payé par billet de toutes les ventes :

```
select min(pricepaid/qtysold)as min_ticket_price
from sales;

min_ticket_price
------------------
20.00000000
(1 row)
```

Fonction PERCENTILE_CONT

La fonction PERCENTILE_CONT est une fonction de distribution inverse qui suppose un modèle de distribution continue. Elle prend une valeur de centile et une spécification de tri, et renvoie une valeur interpolée qui entre dans la catégorie de la valeur de centile donnée en ce qui concerne la spécification de tri.

PERCENTILE_CONT calcule une interpolation linéaire entre les valeurs après les avoir ordonnées. A l'aide de la valeur de centile `(P)` et le nombre de lignes non null `(N)` dans le groupe d'agrégation, la fonction calcule le nombre de lignes après l'ordonnancement des lignes en fonction de la spécification de tri. Ce nombre de lignes `(RN)` est calculé selon la formule `RN = (1+ (P*(N-1))`.

Le résultat de la fonction d'agrégation est calculé par interpolation linéaire entre les valeurs des lignes aux numéros de ligne `CRN = CEILING(RN)` et `FRN = FLOOR(RN)`.

Le résultat final sera le suivant.

Si `(CRN = FRN = RN)` le résultat est `(value of expression from row at RN)`

Sinon, le résultat est le suivant :

`(CRN - RN) * (value of expression for row at FRN) + (RN - FRN) * (value of expression for row at CRN)`.

Syntaxe

```
PERCENTILE_CONT(percentile)
WITHIN GROUP(ORDER BY expr)
```

Arguments

percentile

Constante numérique comprise entre 0 et 1. Les valeurs `NULL` sont ignorées dans le calcul.

expr

Spécifie date/time les valeurs numériques ou sur lesquelles trier et calculer le percentile.

Renvoie

Le type de retour est déterminé par le type de données de l'expression ORDER BY dans la clause WITHIN GROUP. Le tableau suivant illustre le type de retour de chaque type de données d'expression ORDER BY.

Type d'entrée	Type de retour
`INT2`, `INT4`, `INT8`, `NUMERIC`, `DECIMAL`	`DECIMAL`
`FLOAT`, `DOUBLE`	`DOUBLE`
`DATE`	`DATE`
`TIMESTAMP`	`TIMESTAMP`

Type d'entrée	Type de retour
TIMESTAMPTZ	TIMESTAMPTZ

Notes d'utilisation

Si l'expression ORDER BY est un type de données DECIMAL défini avec la précision maximale de 38 chiffres, il est possible que PERCENTILE_CONT renvoie un résultat inexact ou une erreur. Si la valeur de retour de la fonction PERCENTILE_CONT dépasse 38 chiffres, le résultat est tronqué pour s'adapter, ce qui entraîne une perte de précision.. Si, au cours de l'interpolation, un résultat intermédiaire dépasse la précision maximale, un dépassement de capacité numérique se produit et la fonction renvoie une erreur. Pour éviter ces conditions, nous vous recommandons d'utiliser un type de données avec une précision inférieure ou l'expression ORDER BY avec une précision inférieure.

Si une instruction inclut plusieurs appels à des fonctions d'agrégation basées sur le tri (LISTAGG, PERCENTILE_CONT, or MEDIAN), elles doivent toutes utiliser les mêmes valeurs ORDER BY. Notez que MEDIAN applique une clause order by implicite sur la valeur d'expression.

Par exemple, l'instruction suivante renvoie une erreur.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

An error occurred when executing the SQL command:
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(pricepaid)
FROM sales
GROUP BY salesid, pricepaid;

ERROR: within group ORDER BY clauses for aggregate functions must be the same
```

L'instruction suivante s'exécute avec succès.

```
SELECT TOP 10 salesid, SUM(pricepaid),
PERCENTILE_CONT(0.6) WITHIN GROUP(ORDER BY salesid),
MEDIAN(salesid)
```

```
FROM sales
GROUP BY salesid, pricepaid;
```

Exemples

Les exemples suivants utilisent l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

L'exemple suivant montre que PERCENTILE_CONT(0.5) produit les mêmes résultats que MEDIAN.

```
SELECT TOP 10 DISTINCT sellerid, qtysold,
PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY qtysold),
MEDIAN(qtysold)
FROM sales
GROUP BY sellerid, qtysold;

+----------+---------+-----------------+--------+
| sellerid | qtysold | percentile_cont | median |
+----------+---------+-----------------+--------+
|        2 |       2 |               2 |      2 |
|       26 |       1 |               1 |      1 |
|       33 |       1 |               1 |      1 |
|       38 |       1 |               1 |      1 |
|       43 |       1 |               1 |      1 |
|       48 |       2 |               2 |      2 |
|       48 |       3 |               3 |      3 |
|       77 |       4 |               4 |      4 |
|       85 |       4 |               4 |      4 |
|       95 |       2 |               2 |      2 |
+----------+---------+-----------------+--------+
```

L'exemple suivant recherche PERCENTILE_CONT(0.5) et PERCENTILE_CONT(0.75) pour la quantité vendue pour chaque ID de vendeur dans la table SALES.

```
SELECT sellerid,
PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY qtysold) as pct_50,
PERCENTILE_CONT(0.75) WITHIN GROUP(ORDER BY qtysold) as pct_75
FROM sales
GROUP BY sellerid
ORDER BY sellerid
LIMIT 10;

+----------+--------+---------+
```

```
| sellerid | pct_50 | pct_75 |
+----------+--------+---------+
|        1 |    1.5 |    1.75 |
|        2 |      2 |    2.25 |
|        3 |      2 |       3 |
|        4 |      2 |       2 |
|        5 |      1 |     1.5 |
|        6 |      1 |       1 |
|        7 |    1.5 |    1.75 |
|        8 |      1 |       1 |
|        9 |      4 |       4 |
|       12 |      2 |    3.25 |
+----------+--------+---------+
```

Pour vérifier les résultats de la requête précédente pour le premier ID de vendeur, utilisez l'exemple suivant.

```
SELECT qtysold
FROM sales
WHERE sellerid=1;

+---------+
| qtysold |
+---------+
|       2 |
|       1 |
+---------+
```

Fonctions STDDEV_SAMP et STDDEV_POP

Les fonctions STDDEV_SAMP et STDDEV_POP renvoient l'écart type entre l'échantillon et la population d'un ensemble de valeurs numériques (nombre entier, décimale ou à virgule flottante). Le résultat de la fonction STDDEV_SAMP est équivalent à la racine carré de la variance de l'échantillon du même ensemble de valeurs.

STDDEV_SAMP et STDDEV sont des synonymes de la même fonction.

Syntaxe

```
STDDEV_SAMP | STDDEV ( [ DISTINCT | ALL ] expression)
STDDEV_POP ( [ DISTINCT | ALL ] expression)
```

L'expression doit comporter un type de données de nombre entier, décimale ou à virgule flottante. Quel que soit le type de données de l'expression, le type de retour de cette fonction est un nombre double précision.

> **Note**
>
> L'écart type est calculé à l'aide de l'arithmétique à virgule flottante, qui peut se traduire par une légère imprécision.

Notes d'utilisation

Lorsque l'écart type de l'échantillon (STDDEV ou STDDEV_SAMP) est calculé pour une expression qui se compose d'une seule valeur, le résultat de la fonction est NULL, pas 0.

Exemples

La requête suivante renvoie la moyenne des valeurs de la colonne VENUESEATS de la table VENUE, suivie par l'écart type de l'échantillon et l'écart type de la population du même ensemble de valeurs. VENUESEATS est une colonne INTEGER. L'échelle du résultat est réduite à 2 chiffres.

```
select avg(venueseats),
cast(stddev_samp(venueseats) as dec(14,2)) stddevsamp,
cast(stddev_pop(venueseats) as dec(14,2)) stddevpop
from venue;

avg  | stddevsamp | stddevpop
-------+------------+-----------
17503 |   27847.76 |  27773.20
(1 row)
```

La requête suivante renvoie l'écart type de l'échantillon pour la colonne COMMISSION de la table SALES. COMMISSION est une virgule DECIMAL. L'échelle du résultat est réduite à 10 chiffres.

```
select cast(stddev(commission) as dec(18,10))
from sales;

stddev
----------------
130.3912659086
(1 row)
```

La requête suivante convertit l'écart type de l'échantillon de la colonne COMMISSION en un nombre entier.

```
select cast(stddev(commission) as integer)
from sales;

stddev
--------
130
(1 row)
```

La requête suivante renvoie l'écart type de l'échantillon et la racine carré de la variance de l'échantillon pour la colonne COMMISSION. Les résultats de ces calculs sont identiques.

```
select
cast(stddev_samp(commission) as dec(18,10)) stddevsamp,
cast(sqrt(var_samp(commission)) as dec(18,10)) sqrtvarsamp
from sales;

stddevsamp   |  sqrtvarsamp
----------------+----------------
130.3912659086 | 130.3912659086
(1 row)
```

Fonction SUM

La fonction SUM renvoie la somme des valeurs de la colonne d'entrée ou de l'expression. La fonction SUM utilise des valeurs numériques et ignore les valeurs NULL.

Syntaxe

```
SUM ( [ DISTINCT | ALL ] expression )
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère. L'expression est l'un des types de données suivants :

- SMALLINT

- INTEGER
- BIGINT
- NUMERIC
- DECIMAL
- REAL
- DOUBLE PRECISON
- SUPER

DISTINCT | ALL

Avec l'argument DISTINCT, la fonction supprime toutes les valeurs en double dans l'expression spécifiée avant de calculer la somme. Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression pour calculer la somme. La valeur par défaut est ALL.

Types de données

Les types d'argument pris en charge par la fonction SUM sont SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL, DOUBLE PRECISION et SUPER.

Les types de retour pris en charge par la fonction SUM sont les suivants :

- BIGINT pour les arguments BIGINT, SMALLINT et INTEGER
- NUMERIC pour les arguments NUMERIC
- DOUBLE PRECISION pour les arguments à virgule flottante
- Renvoie le même type de données que l'expression pour tout autre type d'argument.

La précision par défaut d'un résultat de fonction SUM avec un argument NUMERIC ou DECIMAL est de 38. L'échelle du résultat est identique à celle de l'argument. Par exemple, une fonction SUM d'une colonne DEC(5,2) renvoie un type de données DEC(38,2).

Exemples

Pour rechercher toutes les commissions payées à partir de la table SALES :

```
select sum(commission) from sales;

sum
-------------
```

```
16614814.65
(1 row)
```

Pour trouver le nombre de places de tous les lieux de l'état de Floride :

```
select sum(venueseats) from venue
where venuestate = 'FL';

sum
--------
250411
(1 row)
```

Pour trouver le nombre de places vendu en mai :

```
select sum(qtysold) from sales, date
where sales.dateid = date.dateid and date.month = 'MAY';

sum
-------
32291
(1 row)
```

Fonctions VAR_SAMP et VAR_POP

Les fonctions VAR_SAMP et VAR_POP renvoient la variance entre l'échantillon et la population d'un ensemble de valeurs numériques (nombre entier, décimale ou à virgule flottante). Le résultat de la fonction VAR_SAMP est équivalent au carré de l'écart type de l'échantillon du même ensemble de valeurs.

VAR_SAMP et VARIANCE sont des synonymes de la même fonction.

Syntaxe

```
VAR_SAMP | VARIANCE ( [ DISTINCT | ALL ] expression)
VAR_POP ( [ DISTINCT | ALL ] expression)
```

L'expression doit comporter un type de données de nombre entier, décimale ou à virgule flottante. Quel que soit le type de données de l'expression, le type de retour de cette fonction est un nombre double précision.

> **Note**
>
> Les résultats de ces fonctions peuvent varier entre les clusters d'entrepôts des données, en fonction de la configuration du cluster dans chaque cas.

Notes d'utilisation

Lorsque l'écart type de l'échantillon (VARIANCE ou VAR_SAMP) est calculé pour une expression qui se compose d'une valeur unique, le résultat de la fonction est NULL pas 0.

Exemples

La requête suivante renvoie la variance arrondie entre l'échantillon et la population de la colonne NUMTICKETS dans la table LISTING.

```
select avg(numtickets),
round(var_samp(numtickets)) varsamp,
round(var_pop(numtickets)) varpop
from listing;

avg | varsamp | varpop
-----+---------+--------
10 |      54 |     54
(1 row)
```

La requête suivante exécute les mêmes calculs mais traduit les résultats en valeur décimales.

```
select avg(numtickets),
cast(var_samp(numtickets) as dec(10,4)) varsamp,
cast(var_pop(numtickets) as dec(10,4)) varpop
from listing;

avg | varsamp | varpop
-----+---------+---------
10 | 53.6291 | 53.6288
(1 row)
```

Fonctions de tableau

Vous trouverez ci-dessous une description des fonctions de tableau pour SQL prises en charge par Amazon Redshift pour accéder aux tableaux et les manipuler.

Rubriques

- Fonction array
- Fonction array_concat
- Fonction array_flatten
- Fonction get_array_length
- Fonction split_to_array
- Fonction subarray

Fonction array

Crée un tableau du type de données SUPER.

Syntaxe

```
ARRAY( [ expr1 ] [ , expr2 [ , ... ] ] )
```

Argument

expr1, expr2

Expressions de n'importe quel type de données Amazon Redshift à l'exception des types de date et d'heure, car Amazon Redshift ne convertit pas les types de date et d'heure en type de données SUPER. Les arguments ne doivent pas nécessairement être du même type de données.

Type de retour

La fonction de tableau renvoie le type de données SUPER.

exemple

Les exemples suivants montrent un tableau de valeurs numériques et un tableau de différents types de données.

```
--an array of numeric values
select array(1,50,null,100);
      array
------------------
 [1,50,null,100]
(1 row)

--an array of different data types
select array(1,'abc',true,3.14);
        array
-----------------------
 [1,"abc",true,3.14]
(1 row)
```

Fonction array_concat

La fonction array_concat concatène deux tableaux pour créer un tableau qui contient tous les éléments du premier tableau suivi de tous les éléments du second tableau. Les deux arguments doivent être des tableaux valides.

Syntaxe

```
array_concat( super_expr1,  super_expr2 )
```

Arguments

super_expr1

Valeur qui spécifie le premier des deux tableaux à concaténer.

super_expr2

Valeur qui spécifie le deuxième des deux tableaux à concaténer.

Type de retour

La fonction array_concat renvoie une valeur de données SUPER.

exemple

Les exemples suivants montrent la concaténation de deux tableaux du même type et la concaténation de deux tableaux de types différents.

```
-- concatenating two arrays
SELECT ARRAY_CONCAT(ARRAY(10001,10002),ARRAY(10003,10004));
              array_concat
-----------------------------------
 [10001,10002,10003,10004]
(1 row)

-- concatenating two arrays of different types
SELECT ARRAY_CONCAT(ARRAY(10001,10002),ARRAY('ab','cd'));
          array_concat
------------------------------
 [10001,10002,"ab","cd"]
(1 row)
```

Fonction array_flatten

Fusionne plusieurs tableaux en un seul tableau de type SUPER.

Syntaxe

```
array_flatten( super_expr1,super_expr2,.. )
```

Arguments

super_expr1,super_expr2

Expression SUPER valide de forme de table.

Type de retour

La fonction array_flatten renvoie une valeur de données SUPER.

exemple

Voici un exemple de la fonction array_flatten.

```
SELECT ARRAY_FLATTEN(ARRAY(ARRAY(1,2,3,4),ARRAY(5,6,7,8),ARRAY(9,10)));
     array_flatten
-----------------------
 [1,2,3,4,5,6,7,8,9,10]
```

```
(1 row)
```

Fonction get_array_length

Renvoie la longueur du tableau spécifié. La fonction GET_ARRAY_LENGTH renvoie la longueur d'un tableau SUPER recevant un chemin d'objet ou de tableau.

Syntaxe

```
get_array_length( super_expr )
```

Arguments

super_expr

Expression SUPER valide de forme de table.

Type de retour

La fonction get_array_length renvoie un BIGINT.

exemple

L’exemple suivant présente une fonction get_array_length.

```
SELECT GET_ARRAY_LENGTH(ARRAY(1,2,3,4,5,6,7,8,9,10));
 get_array_length
----------------------
            10
(1 row)
```

Fonction split_to_array

Utilise un délimiteur en tant que paramètre facultatif. Si aucun délimiteur n’est défini, la valeur par défaut est une virgule.

Syntaxe

```
split_to_array( string,delimiter )
```

Arguments

string

Chaîne d'entrée à fractionner.

delimiter

Valeur facultative sur laquelle la chaîne d'entrée sera fractionnée. La valeur par défaut est une virgule.

Type de retour

La fonction split_to_array renvoie une valeur de données SUPER.

exemple

L'exemple suivant montre une fonction split_to_array.

```
SELECT SPLIT_TO_ARRAY('12|345|6789', '|');
     split_to_array
------------------------
 ["12","345","6789"]
(1 row)
```

Fonction subarray

Manipule les tableaux pour renvoyer un sous-ensemble des tableaux d'entrée.

Syntaxe

```
SUBARRAY( super_expr, start_position, length )
```

Arguments

super_expr

Expression SUPER valide sous forme de tableau.

start_position

Position dans le tableau à partir de laquelle commence l'extraction, soit la position d'index 0. Une position négative signifie que l'extraction se fait à l'envers, en partant de la fin du tableau.

longueur

Nombre d'éléments à extraire (longueur de la sous-chaîne).

Type de retour

La fonction subarray renvoie une valeur de données SUPER.

Exemples

Voici un exemple de sortie de fonction subarray :

```
 SELECT SUBARRAY(ARRAY('a', 'b', 'c', 'd', 'e', 'f'), 2, 3);
   subarray
---------------
 ["c","d","e"]
(1 row)
```

Fonctions d'agrégation bit par bit

Les fonctions d'agrégation bit par bit calculent les opérations binaires pour effectuer l'agrégation des colonnes d'entiers et des colonnes pouvant être converties ou arrondies en valeurs entières.

Rubriques

- Utilisation NULLs dans des agrégations au niveau du bit
- Prise en charge de la fonction DISTINCT pour les agrégations bit par bit
- Exemples de présentation pour les fonctions bit par bit
- Fonction BIT_AND
- Fonction BIT_OR
- Fonction BOOL_AND
- Fonction BOOL_OR

Utilisation NULLs dans des agrégations au niveau du bit

Lorsque vous appliquez une fonction bit par bit à une colonne autorisant les valeurs nulles, toutes les valeurs NULL sont éliminées avant le calcul du résultat de la fonction. Si aucune ligne n'est admissible à l'agrégation, la fonction bit par bit renvoie NULL. Le même comportement s'applique aux fonctions d'agrégation standard. Voici un exemple.

```
select sum(venueseats), bit_and(venueseats) from venue
where venueseats is null;

sum  | bit_and
------+---------
null |    null
(1 row)
```

Prise en charge de la fonction DISTINCT pour les agrégations bit par bit

Tout comme d'autres fonctions d'agrégation, les fonctions bit par bit prennent en charge le mot-clé DISTINCT.

Toutefois, l'utilisation de DISTINCT avec ces fonctions n'a aucun impact sur les résultats. La première instance d'une valeur est suffisante pour satisfaire des opérations AND ou OR bit par bit. Cela ne fait aucune différence si des valeurs en double sont présentes dans l'expression qui est évaluée.

Du fait que le traitement DISTINCT risque d'entraîner une certaine surcharge de l'exécution des requêtes, nous vous recommandons de ne pas utiliser DISTINCT avec les fonctions bit par bit.

Exemples de présentation pour les fonctions bit par bit

Vous trouverez ci-dessous quelques exemples de présentation montrant comment utiliser les fonctions bit par bit. Vous trouverez également des exemples de code spécifiques avec la description de chaque fonction.

Les exemples de fonctions bit par bit sont basés sur l'exemple de base de données TICKIT. La table USERS de l'exemple de base de données TICKIT contient plusieurs colonnes booléennes qui indiquent si chaque utilisateur est connu pour aimer différents types d'événements, comme le sport, le théâtre, l'opéra et ainsi de suite. Un exemple suit.

```
select userid, username, lastname, city, state,
likesports, liketheatre
from users limit 10;

userid | username | lastname  |     city     | state | likesports | liketheatre
-------+----------+-----------+--------------+-------+------------+-------------
1 | JSG99FHE | Taylor    | Kent         | WA    | t          | t
9 | MSD36KVR | Watkins   | Port Orford  | MD    | t          | f
```

Supposons qu'une nouvelle version de la table USERS est construite différemment. Dans cette nouvelle version, une colonne à un seul entier définit (sous forme binaire) huit types d'événements que chaque utilisateur aime ou n'aime pas. Dans cette conception, chaque position de bit représente un type d'événement. Un utilisateur qui aime les huit types d'évènements a chacun d'eux défini sur 1 (comme à la première ligne du tableau suivant). Un utilisateur qui n'aime aucun de ces événements a les huit bits définis sur 0 (voir la deuxième ligne). Un utilisateur qui aime uniquement le sport et le jazz est représenté sur la troisième ligne suivante.

USER	SPORTS	THEATRE	JAZZ	OPERA	ROCK	VEGAS	BROADW	CLASSICAL
User 1	1	1	1	1	1	1	1	1
User 2	0	0	0	0	0	0	0	0
User 3	1	0	1	0	0	0	0	0

Dans la table de base de données, ces valeurs binaires peuvent être stockées dans une seule colonne LIKES sous forme d'entiers, comme illustré ci-dessous.

Utilisateur	Valeur binaire	Valeur stockée (nombre entier)
User 1	11111111	255
User 2	00000000	0
User 3	10100000	160

Fonction BIT_AND

La fonction BIT_AND exécute des opérations AND bit par bit sur toutes les valeurs d'une colonne ou expression d'entiers. Ces fonctions regroupent chaque bit de chaque valeur binaire correspondant à chaque valeur de nombre entier de l'expression.

La fonction BIT_AND renvoie un résultat de `0` si aucun des bits n'est défini sur 1 dans l'ensemble des valeurs. Si un ou plusieurs bits est défini sur 1 dans toutes les valeurs, la fonction renvoie une valeur de nombre entier. Ce nombre entier est le chiffre qui correspond à la valeur binaire de ces bits.

Par exemple, une table contient quatre valeurs de nombre entier dans une colonne : 3, 7, 10 et 22. Ces nombres entiers sont représentés sous la forme binaire suivante :

Entier	Valeur binaire
3	11
7	111
10	1010
22	10110

Une opération BIT_AND sur cet ensemble de données détecte que tous les bits sont définis sur 1 la second-to-last position uniquement. Le résultat est une valeur binaire de `00000010`, qui représente la valeur d'entier 2. Par conséquent, la fonction BIT_AND renvoie 2.

Syntaxe

```
BIT_AND ( [DISTINCT | ALL] expression )
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère. Cette expression doit avoir un INT ou un type de INT8 données. INT2 La fonction renvoie un INT ou un type de INT8 données équivalent. INT2

DISTINCT | ALL

Avec l'argument DISTINCT, la fonction supprime toutes les valeurs en double de l'expression spécifiée avant de calculer le résultat. Avec l'argument ALL, la fonction conserve toutes les valeurs en double. La valeur par défaut est ALL. Pour plus d'informations, consultez Prise en charge de la fonction DISTINCT pour les agrégations bit par bit.

Exemples

Étant donné que des informations importantes sur l'entreprise sont stockées dans les colonnes de nombres entiers, vous pouvez utiliser des fonctions bit par bit pour extraire et regrouper ces

informations. La requête suivante applique la fonction BIT_AND à la colonne LIKES dans une table appelée USERLIKES et regroupe les résultats en fonction de la colonne CITY.

```
select city, bit_and(likes) from userlikes group by city
order by city;
city          | bit_and
--------------+---------
Los Angeles   |       0
Sacramento    |       0
San Francisco |       0
San Jose      |      64
Santa Barbara |     192
(5 rows)
```

Vous pouvez interpréter ces résultats comme suit :

- La valeur de nombre entier 192 de Santa Barbara se traduit par la valeur binaire `11000000`. Autrement dit, tous les utilisateurs de cette ville aiment le sport et le théâtre, mais tous les utilisateurs n'aiment pas un tout autre type d'événement.
- Le nombre entier 64 se traduit par `01000000`. Ainsi, pour les utilisateurs de San Jose, le seul type d'événement qu'ils aiment tous est le théâtre.
- Les valeurs de `0` des trois autres villes indiquent qu'aucun « J'aime » n'est partagé par tous les utilisateurs de ces villes.

Fonction BIT_OR

La fonction BIT_OR exécute des opérations OR bit par bit sur toutes les valeurs d'une seule colonne ou expression d'entiers. Ces fonctions regroupent chaque bit de chaque valeur binaire correspondant à chaque valeur de nombre entier de l'expression.

Par exemple, supposons que votre table contient quatre valeurs de nombre entier dans une colonne : 3, 7, 10 et 22. Ces nombres entiers sont représentés sous la forme binaire suivante.

Entier	Valeur binaire
3	11
7	111

Entier	Valeur binaire
10	1010
22	10110

Si vous appliquez la fonction BIT_OR au même ensemble de valeurs entières, l'opération recherche toutes les valeurs contenant 1 à chaque position. Dans ce cas, il existe une valeur de 1 aux cinq dernières positions d'au moins une des valeurs, ce qui donne entraîne un résultat binaire de `00011111`. Par conséquent, la fonction renvoie `31` (ou `16 + 8 + 4 + 2 + 1`).

Syntaxe

```
BIT_OR ( [DISTINCT | ALL] expression )
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère. Cette expression doit avoir un INT ou un type de INT8 données. INT2 La fonction renvoie un INT ou un type de INT8 données équivalent. INT2

DISTINCT | ALL

Avec l'argument DISTINCT, la fonction supprime toutes les valeurs en double de l'expression spécifiée avant de calculer le résultat. Avec l'argument ALL, la fonction conserve toutes les valeurs en double. La valeur par défaut est ALL. Pour plus d'informations, consultez Prise en charge de la fonction DISTINCT pour les agrégations bit par bit.

exemple

La requête suivante applique la fonction BIT_AND à la colonne LIKES dans une table appelée USERLIKES et regroupe les résultats en fonction de la colonne CITY.

```
select city, bit_or(likes) from userlikes group by city
order by city;
city          | bit_or
--------------+--------
```

```
Los Angeles   |    127
Sacramento    |    255
San Francisco |    255
San Jose      |    255
Santa Barbara |    255
(5 rows)
```

Pour quatre des villes répertoriées, tous les types d'événements sont appréciés par au moins un utilisateur (`255=11111111`). Pour Los Angeles, tous les types d'événements, sauf le sport sont appréciés par au moins un utilisateur (`127=01111111`).

Fonction BOOL_AND

La fonction BOOL_AND opère sur une seule colonne ou expression booléenne ou entière. Elle applique une logique similaire aux fonctions BIT_AND et BIT_OR. Pour cette fonction, le type de retour est une valeur booléenne (`true` ou `false`).

Si toutes les valeurs d'un ensemble sont true, la fonction BOOL_AND renvoie `true` (`t`). Si une valeur est false, la fonction renvoie `false` (`f`).

Syntaxe

```
BOOL_AND ( [DISTINCT | ALL] expression )
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère. Cette expression doit comporter un type de données BOOLEAN ou nombre entier. Le type de retour de la fonction est BOOLEAN.

DISTINCT | ALL

Avec l'argument DISTINCT, la fonction supprime toutes les valeurs en double de l'expression spécifiée avant de calculer le résultat. Avec l'argument ALL, la fonction conserve toutes les valeurs en double. La valeur par défaut est ALL. Pour plus d'informations, consultez Prise en charge de la fonction DISTINCT pour les agrégations bit par bit.

Exemples

Vous pouvez utiliser les fonctions booléennes par rapport à des expressions booléennes ou à des expressions de type nombre entier. Par exemple, la requête suivante renvoie les résultats de la table USERS standard de la base de données TICKIT, qui comporte plusieurs colonnes booléennes.

La fonction BOOL_AND renvoie `false` pour les cinq lignes. Tous les utilisateurs de chacun de ces états n'aiment pas le sport.

```
select state, bool_and(likesports) from users
group by state order by state limit 5;

state | bool_and
------+---------
AB    | f
AK    | f
AL    | f
AZ    | f
BC    | f
(5 rows)
```

Fonction BOOL_OR

La fonction BOOL_OR opère sur une seule colonne ou expression booléenne ou entière. Elle applique une logique similaire aux fonctions BIT_AND et BIT_OR. Pour cette fonction, le type de retour est une valeur booléenne (`true`, `false` ou `NULL`).

Si une ou plusieurs valeurs d'un ensemble le sont`true`, la fonction BOOL_OR renvoie `true` (). `t` Si toutes les valeurs d'un ensemble le sont`false`, la fonction renvoie `false` (`f`). La valeur NULL peut être renvoyée si la valeur est inconnue.

Syntaxe

```
BOOL_OR ( [DISTINCT | ALL] expression )
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère. Cette expression doit comporter un type de données BOOLEAN ou nombre entier. Le type de retour de la fonction est BOOLEAN.

DISTINCT | ALL

Avec l'argument DISTINCT, la fonction supprime toutes les valeurs en double de l'expression spécifiée avant de calculer le résultat. Avec l'argument ALL, la fonction conserve toutes les valeurs en double. La valeur par défaut est ALL. Consultez Prise en charge de la fonction DISTINCT pour les agrégations bit par bit.

Exemples

Vous pouvez utiliser les fonctions booléennes avec des expressions booléennes ou des expressions de type nombre entier. Par exemple, la requête suivante renvoie les résultats de la table USERS standard de la base de données TICKIT, qui comporte plusieurs colonnes booléennes.

La fonction BOOL_OR renvoie `true` pour les cinq lignes. Au moins un utilisateur de chacun de ces états aime le sport.

```
select state, bool_or(likesports) from users
group by state order by state limit 5;

state | bool_or
------+--------
AB    | t
AK    | t
AL    | t
AZ    | t
BC    | t
(5 rows)
```

L'exemple suivant renvoie la valeur NULL.

```
SELECT BOOL_OR(NULL = '123')
               bool_or
------
NULL
```

Expressions conditionnelles

Rubriques

- Expression conditionnelle CASE
- Fonction DECODE

- Fonctions GREATEST et LEAST
- Fonctions NVL et COALESCE
- NVL2 fonction
- Fonction NULLIF

Amazon Redshift prend en charge des expressions conditionnelles qui sont des extensions de la norme SQL.

Expression conditionnelle CASE

L'expression CASE est une expression conditionnelle, similaire aux if/then/else instructions trouvées dans d'autres langages. L'expression CASE est utilisée pour spécifier un résultat lorsqu'il y a plusieurs conditions. Utilisez CASE là où l'utilisation d'une expression SQL est valide, par exemple dans une commande SELECT.

Il existe deux types d'expressions CASE : simple et recherchée.

- Dans les expressions CASE simples, une expression est comparée à une valeur. Lorsqu'une correspondance est trouvée, l'action spécifiée dans la clause THEN est appliquée. Si aucune correspondance n'est trouvée, l'action de la clause ELSE est appliquée.
- Dans les expressions CASE recherchées, chaque expression CASE est évaluée en fonction d'une expression booléenne, et l'instruction CASE renvoie la première expression CASE correspondante. Si aucune correspondance n'est trouvée parmi les clauses WHEN, l'action contenue dans la clause ELSE est renvoyée.

Syntaxe

Instruction CASE simple utilisée pour mettre en correspondance des conditions :

```
CASE expression
  WHEN value THEN result
  [WHEN...]
  [ELSE result]
END
```

Instructions CASE recherchées utilisées pour évaluer chaque condition :

```
CASE
```

```
  WHEN condition THEN result
  [WHEN ...]
  [ELSE result]
END
```

Arguments

expression

Nom de la colonne ou n'importe quelle expression valide.

valeur

Valeur à laquelle l'expression est comparée, par exemple une constante numérique ou une chaîne de caractères.

result

Valeur ou expression cible qui est renvoyée lorsqu'une expression ou une condition booléenne est évaluée. Les types de données de toutes les expressions de résultat doivent pouvoir être convertis en un seul type de sortie.

condition

Expression booléenne qui prend la valeur true ou false. Si la condition a la valeur true, la valeur de l'expression CASE est le résultat qui suit la condition, et le reste de l'expression CASE n'est pas traité. Si la condition a la valeur false, les clauses WHEN suivantes sont évaluées. Si aucune condition WHEN n'a la valeur true en résultat, la valeur de l'expression CASE est le résultat de la clause ELSE. Si la clause ELSE est omise et qu'aucune condition n'a la valeur true, le résultat est null.

Exemples

Les exemples suivants utilisent la table VENUE et la table SALES de l'exemple de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Utilisez une expression CASE simple pour remplacer `New York City` par `Big Apple` dans une requête sur la table de VENUE. Remplacer tous les autres noms de villes par `other`.

```
select venuecity,
  case venuecity
    when 'New York City'
    then 'Big Apple' else 'other'
```

```
  end
from venue
order by venueid desc;

venuecity        |   case
-----------------+-----------
Los Angeles      | other
New York City    | Big Apple
San Francisco    | other
Baltimore        | other
...
```

Utiliser une expression CASE recherchée pour affecter des numéros de groupes basés sur la valeur PRICEPAID pour les vente de billets individuelles :

```
select pricepaid,
  case when pricepaid <10000 then 'group 1'
    when pricepaid >10000 then 'group 2'
    else 'group 3'
  end
from sales
order by 1 desc;

pricepaid |  case
----------+---------
12624     | group 2
10000     | group 3
10000     | group 3
9996      | group 1
9988      | group 1
...
```

Fonction DECODE

Une expression DECODE remplace une valeur spécifique par une autre valeur spécifique ou une valeur par défaut, selon le résultat d'une condition d'égalité. Cette opération est équivalente à l'opération d'une simple expression CASE ou d'une IF-THEN-ELSE instruction.

Syntaxe

```
DECODE ( expression, search, result [, search, result ]... [ ,default ] )
```

Ce type d’expression est utile pour remplacer des abréviations ou des codes stockés dans les tables par des valeurs commerciales pertinentes qui sont nécessaires pour les rapports.

Paramètres

expression

Source de la valeur que vous souhaitez comparer, comme une colonne dans une table.

search

Valeur cible comparée à l’expression source, par exemple une valeur numérique ou une chaîne de caractères. L’expression search doit correspondre à une seule valeur fixe. Vous ne pouvez pas spécifier une expression qui correspond à une plage de valeurs, par exemple `age between 20 and 29` ; vous devez spécifier des search/result paires distinctes pour chaque valeur que vous souhaitez remplacer.

Le type de données de toutes les instances de l’expression search doit être identique ou compatible. Les paramètres expression et search doivent aussi être compatibles.

result

Valeur de remplacement que la requête renvoie lorsque l’expression correspond à la valeur de recherche. Vous devez inclure au moins une search/result paire dans l'expression DECODE.

Les types de données de toutes les instances de l’expression result doivent être identiques ou compatibles. Les paramètres result et default doivent aussi être compatibles.

default

Valeur par défaut d’une option utilisée pour les cas où la condition de recherche échoue. Si vous ne spécifiez pas de valeur par défaut, l’expression DECODE renvoie NULL.

Notes d’utilisation

Si expression et search ont la valeur NULL, le résultat DECODE est la valeur result correspondante. Pour une illustration de cette utilisation de la fonction, consultez Exemples.

Lorsqu’il est utilisé de cette façon, DECODE est similaire à NVL2 fonction, mais il existe quelques différences. Pour une description de ces différences, consultez les notes NVL2 d'utilisation.

Exemples

Lorsque la valeur `2008-06-01` existe dans la colonne CALDATE de DATETABLE, l'exemple suivant la remplace par `June 1st, 2008`. L'exemple remplace toutes les autres valeurs CALDATE par NULL.

```
select decode(caldate, '2008-06-01', 'June 1st, 2008')
from datetable where month='JUN' order by caldate;

case
----------------
June 1st, 2008

...
(30 rows)
```

L'exemple suivant utilise une expression DECODE pour convertir les cinq colonnes CATNAME abrégées dans la table CATEGORY en noms complets et convertir les autres valeurs de la colonne en `Unknown`.

```
select catid, decode(catname,
'NHL', 'National Hockey League',
'MLB', 'Major League Baseball',
'MLS', 'Major League Soccer',
'NFL', 'National Football League',
'NBA', 'National Basketball Association',
'Unknown')
from category
order by catid;

catid  | case
-------+---------------------------------
1      | Major League Baseball
2      | National Hockey League
3      | National Football League
4      | National Basketball Association
5      | Major League Soccer
6      | Unknown
7      | Unknown
8      | Unknown
9      | Unknown
10     | Unknown
```

```
11     | Unknown
(11 rows)
```

Utilisez une expression DECODE pour rechercher des sites au Colorado et au Nevada avec NULL dans la colonne VENUESEATS ; convertissez-le en zéros. NULLs Si la colonne VENUESEATS n'est pas NULL, renvoyez 1 comme résultat.

```
select venuename, venuestate, decode(venueseats,null,0,1)
from venue
where venuestate in('NV','CO')
order by 2,3,1;

venuename                | venuestate     | case
-----------------------------+----------------+-----------
Coors Field                  | CO        |   1
Dick's Sporting Goods Park   | CO        |   1
Ellie Caulkins Opera House   | CO        |   1
INVESCO Field        | CO        |   1
Pepsi Center        | CO        |   1
Ballys Hotel        | NV        |   0
Bellagio Hotel               | NV        |   0
Caesars Palace               | NV        |   0
Harrahs Hotel                | NV        |   0
Hilton Hotel                 | NV        |   0
...
(20 rows)
```

Fonctions GREATEST et LEAST

Renvoie la valeur la plus grande ou la plus petite d'une liste d'un nombre quelconque d'expressions.

Syntaxe

```
GREATEST (value [, ...])
LEAST (value [, ...])
```

Paramètres

expression_list

Liste d'expressions séparées par des virgules, telles que des noms de colonnes. Les expressions doivent toutes être converties dans un type de données commun. Les valeurs NULL de la liste sont ignorées. Si toutes les expressions sont évaluées à NULL, le résultat est NULL.

Renvoie

Renvoie la plus grande (pour GREATEST) ou la plus petite (pour LEAST) valeur de la liste d'expressions fournie.

exemple

L'exemple suivant renvoie la valeur la plus élevée dans l'ordre alphabétique pour `firstname` ou `lastname`.

```
select firstname, lastname, greatest(firstname,lastname) from users
where userid < 10
order by 3;

 firstname | lastname  | greatest
-----------+-----------+-----------
 Lars      | Ratliff   | Ratliff
 Reagan    | Hodge     | Reagan
 Colton    | Roy       | Roy
 Barry     | Roy       | Roy
 Tamekah   | Juarez    | Tamekah
 Rafael    | Taylor    | Taylor
 Victor    | Hernandez | Victor
 Vladimir  | Humphrey  | Vladimir
 Mufutau   | Watkins   | Watkins
(9 rows)
```

Fonctions NVL et COALESCE

Renvoie la valeur de la première expression qui n'est pas nulle dans une série d'expressions. Lorsqu'une valeur non nulle est trouvée, les expressions restantes de la liste ne sont pas évaluées.

NVL est identique à COALESCE. Ce sont des synonymes. Cette rubrique explique la syntaxe et contient des exemples pour les deux.

Syntaxe

```
NVL( expression, expression, ... )
```

La syntaxe de COALESCE est identique :

```
COALESCE( expression, expression, ... )
```

Si toutes les expressions régulières sont null, le résultat est null.

Ces fonctions sont utiles lorsque vous souhaitez renvoyer une valeur secondaire lorsqu'une valeur primaire est manquante ou nulle. Par exemple, une requête peut renvoyer le premier des trois numéros de téléphone disponibles : portable, domicile ou travail. L'ordre des expressions dans la fonction détermine l'ordre d'évaluation.

Arguments

expression

Expression, telle qu'un nom de colonne, à évaluer pour l'état null.

Type de retour

Amazon Redshift détermine le type de données de la valeur renvoyée en fonction des expressions d'entrée. Si les types de données des expressions d'entrée n'ont pas de type commun, une erreur est renvoyée.

Exemples

Si la liste contient des expressions entières, la fonction renvoie un entier.

```
SELECT COALESCE(NULL, 12, NULL);

coalesce
--------------
12
```

Cet exemple, qui est identique à l'exemple précédent, sauf qu'il utilise NVL, renvoie le même résultat.

```
SELECT NVL(NULL, 12, NULL);
```

```
coalesce
--------------
12
```

L'exemple suivant renvoie une chaîne de caractères.

```
SELECT COALESCE(NULL, 'Amazon Redshift', NULL);

coalesce
--------------
Amazon Redshift
```

L'exemple suivant génère une erreur car les types de données varient dans la liste d'expressions. Dans ce cas, la liste contient à la fois un type de chaîne et un type de nombre.

```
SELECT COALESCE(NULL, 'Amazon Redshift', 12);
ERROR: invalid input syntax for integer: "Amazon Redshift"
```

Dans cet exemple, vous créez une table avec les colonnes START_DATE et END_DATE, vous insérez des lignes comprenant des valeurs nulles, puis vous appliquez une expression NVL aux deux colonnes.

```
create table datetable (start_date date, end_date date);
insert into datetable values ('2008-06-01','2008-12-31');
insert into datetable values (null,'2008-12-31');
insert into datetable values ('2008-12-31',null);
```

```
select nvl(start_date, end_date)
from datetable
order by 1;

coalesce
------------
2008-06-01
2008-12-31
2008-12-31
```

Le nom de colonne par défaut pour une expression NVL est COALESCE. La requête suivante renvoie les mêmes résultats :

```
select coalesce(start_date, end_date)
```

```
from datetable
order by 1;
```

Dans les exemples de requêtes suivants, vous créez une table contenant des exemples d'informations sur les réservations d'hôtel et insérez plusieurs lignes. Certains enregistrements contiennent des valeurs nulles.

```
create table booking_info (booking_id int, booking_code character(8), check_in date,
 check_out date, funds_collected numeric(12,2));
```

Insérez l'exemple de données suivant. Certains enregistrements n'ont pas de date `check_out` ou de montant `funds_collected`.

```
insert into booking_info values (1, 'OCEAN_WV', '2023-02-01','2023-02-03',100.00);
insert into booking_info values (2, 'OCEAN_WV', '2023-04-22','2023-04-26',120.00);
insert into booking_info values (3, 'DSRT_SUN', '2023-03-13','2023-03-16',125.00);
insert into booking_info values (4, 'DSRT_SUN', '2023-06-01','2023-06-03',140.00);
insert into booking_info values (5, 'DSRT_SUN', '2023-07-10',null,null);
insert into booking_info values (6, 'OCEAN_WV', '2023-08-15',null,null);
```

La requête suivante renvoie une liste de dates. Si la date `check_out` n'est pas disponible, la date `check_in` est indiquée.

```
select coalesce(check_out, check_in)
from booking_info
order by booking_id;
```

Voici les résultats. Notez que les deux derniers enregistrements indiquent la date `check_in`.

```
coalesce
------------
2023-02-03
2023-04-26
2023-03-16
2023-06-03
2023-07-10
2023-08-15
```

Si vous prévoyez qu'une requête renvoie des valeurs null pour certaines fonctions ou colonnes, vous pouvez utiliser une expression NVL pour remplacer les valeurs NULL par une autre valeur.

Par exemple, les fonctions d'agrégation, telles que SUM, renvoient des valeurs null au lieu de zéros lorsqu'elles n'ont aucune ligne à évaluer. Vous pouvez utiliser une expression NVL pour remplacer ces valeurs null par `700.0`. Au lieu de `485`, la somme de `funds_collected` a pour résultat `1885`, car les deux lignes contenant la valeur null sont remplacées par `700`.

```
select sum(nvl(funds_collected, 700.0)) as sumresult from booking_info;

sumresult
------
 1885
```

NVL2 fonction

Renvoie l'une des deux valeurs selon qu'une expression spécifiée a pour valeur NULL ou NOT NULL.

Syntaxe

```
NVL2 ( expression, not_null_return_value, null_return_value )
```

Arguments

expression

Expression, telle qu'un nom de colonne, à évaluer pour l'état null.

not_null_return_value

Valeur renvoyée si expression a une valeur NOT NULL. La valeur not_null_return_value doit avoir le même type de données que expression ou être implicitement convertie en ce type de données.

null_return_value

Valeur renvoyée si expression a une valeur NULL. La valeur null_return_value doit avoir le même type de données que expression ou être implicitement convertie en ce type de données.

Type de retour

Le type de NVL2 retour est déterminé comme suit :

- Si not_null_return_value ou null_return_value a une valeur null, le type de données de l'expression non-null est renvoyé.

Si not_null_return_value et null_return_value n'ont pas de valeur null :

- Si not_null_return_value et null_return_value ont le même type de données que le type de données renvoyé.
- Si not_null_return_value et null_return_value ont des types de données numériques distincts, le type de données numériques compatible le plus petit est renvoyé.
- Si not_null_return_value et null_return_value ont des types de données datetime distincts, un type de données d'horodatage est renvoyé.
- Si not_null_return_value et null_return_value ont des types de données de caractères distincts, le type de données not_null_return_value est renvoyé.
- Si not_null_return_value et null_return_value ont des types de données numériques et non numériques mixtes, le type de données de not_null_return_value est renvoyé.

> **Important**
>
> Dans les deux derniers cas où le type de données de not_null_return_value est renvoyé, null_return_value est converti implicitement en ce type de données. Si les types de données sont incompatibles, la fonction échoue.

Notes d'utilisation

Fonction DECODEpeut être utilisé de la même manière que NVL2 lorsque l'expression et les paramètres de recherche sont tous deux nuls. La différence est que pour DECODE, le retour comportera à la fois la valeur et le type de données du paramètres result. En revanche, pour NVL2, le retour aura la valeur du paramètre not_null_return_value ou null_return_value, selon ce qui est sélectionné par la fonction, mais aura le type de données not_null_return_value.

Par exemple, en supposant que column1 a la valeur NULL, les requêtes suivantes renverront la même valeur. Cependant, le type de données de la valeur de retour DECODE sera INTEGER et le type de données de la valeur de NVL2 retour sera VARCHAR.

```
select decode(column1, null, 1234, '2345');
select nvl2(column1, '2345', 1234);
```

exemple

L'exemple suivant modifie quelques exemples de données, puis évalue les deux champs pour fournir des informations de contact aux utilisateurs :

```
update users set email = null where firstname = 'Aphrodite' and lastname = 'Acevedo';

select (firstname + ' ' + lastname) as name,
nvl2(email, email, phone) AS contact_info
from users
where state = 'WA'
and lastname  like 'A%'
order by lastname, firstname;

name        contact_info
--------------------+--------------------------------------------
Aphrodite Acevedo (906) 632-4407
Caldwell Acevedo  Nunc.sollicitudin@Duisac.ca
Quinn Adams    vel@adipiscingligulaAenean.com
Kamal Aguilar  quis@vulputaterisusa.com
Samson Alexander  hendrerit.neque@indolorFusce.ca
Hall Alford    ac.mattis@vitaediamProin.edu
Lane Allen     et.netus@risusDonec.org
Xander Allison    ac.facilisis.facilisis@Infaucibus.com
Amaya Alvarado    dui.nec.tempus@eudui.edu
Vera Alvarez   at.arcu.Vestibulum@pellentesque.edu
Yetta Anthony  enim.sit@risus.org
Violet Arnold  ad.litora@at.com
August Ashley  consectetuer.euismod@Phasellus.com
Karyn Austin   ipsum.primis.in@Maurisblanditenim.org
Lucas Ayers    at@elitpretiumet.com
```

Fonction NULLIF

Syntaxe

L'expression NULLIF compare les deux arguments et renvoie la valeur nulle si les arguments sont égaux. Si ce n'est pas le cas, le premier argument est renvoyé. Cette expression est l'inverse de l'expression NVL ou COALESCE.

```
NULLIF ( expression1, expression2 )
```

Arguments

expression1, expression2

Colonnes ou expressions cible qui sont comparées. Le type de retour est le identique au type de la première expression. Le nom de colonne par défaut du résultat NULLIF correspond à celui de la première expression.

Exemples

Dans l'exemple suivant, la requête renvoie la chaîne `first` car les arguments ne sont pas égaux.

```
SELECT NULLIF('first', 'second');

case
-------
first
```

Dans l'exemple suivant, la requête renvoie NULL car les arguments littéraux de la chaîne sont égaux.

```
SELECT NULLIF('first', 'first');

case
-------
NULL
```

Dans l'exemple suivant, la requête renvoie 1 car les arguments entiers ne sont pas égaux.

```
SELECT NULLIF(1, 2);

case
-------
1
```

Dans l'exemple suivant, la requête renvoie NULL car les arguments entiers sont égaux.

```
SELECT NULLIF(1, 1);

case
-------
```

```
NULL
```

Dans l'exemple suivant, la requête renvoie la valeur nulle lorsque les valeurs LISTID et SALESID correspondent :

```
select nullif(listid,salesid), salesid
from sales where salesid<10 order by 1, 2 desc;

listid  | salesid
--------+---------
     4  |       2
     5  |       4
     5  |       3
     6  |       5
     10 |       9
     10 |       8
     10 |       7
     10 |       6
        |       1
(9 rows)
```

Vous pouvez utiliser NULLIF pour vous assurer que les chaînes vides sont toujours renvoyées sous forme de valeurs NULL. Dans l'exemple ci-dessous, l'expression NULLIF renvoie une valeur null ou une chaîne qui contient au moins un caractère.

```
insert into category
values(0,'','Special','Special');

select nullif(catgroup,'') from category
where catdesc='Special';

catgroup
----------
null
(1 row)
```

NULLIF ignore les espaces de fin. Si une chaîne n'est pas vide, mais contient des espaces, NULLIF renvoie toujours null :

```
create table nulliftest(c1 char(2), c2 char(2));
```

```
insert into nulliftest values ('a','a ');

insert into nulliftest values ('b','b');


select nullif(c1,c2) from nulliftest;
c1
------
null
null
(2 rows)
```

Fonctions de formatage des types de données

Rubriques

- Fonction CAST
- Fonction CONVERT
- TEXT_TO_INT_ALT
- TEXT_TO_NUMERIC_ALT
- TO_CHAR
- Fonction TO_DATE
- TO_NUMBER
- Fonction TRY_CAST
- Chaînes de format datetime
- Chaînes de format numériques
- Caractères de formatage de type Teradata pour les données numériques

Les fonctions de formatage des types de données offrent un moyen simple de convertir des valeurs d'un type de données à un autre. Pour chacune de ces fonctions, le premier argument est toujours la valeur à formater et le second argument contient le modèle du nouveau format. Amazon Redshift prend en charge plusieurs fonctions de formatage des types de données.

Fonction CAST

La fonction CAST convertit un type de données en un autre type de données compatible. Par exemple, vous pouvez convertir une chaîne en date ou un type numérique en chaîne. CAST effectue

une conversion d'exécution, ce qui signifie que la conversion ne modifie pas le type de données d'une valeur dans une table source. Elle n'est modifiée que dans le contexte de la requête.

La fonction CAST est très similaire à la fonction the section called "CONVERT", en ce sens qu'elles convertissent toutes deux un type de données en un autre, mais elles sont appelées différemment.

Certains types de données nécessitent une conversion explicite en d'autres types de données à l'aide de la fonction CAST ou CONVERT. D'autres types de données peuvent être convertis implicitement, dans le cadre d'une autre commande, sans utiliser CAST ou CONVERT. Consultez Compatibilité et conversion de types.

Syntaxe

Utilisez l'une de ces deux formes de syntaxes équivalentes pour convertir les expressions cast d'un type de données à un autre.

```
CAST ( expression AS type )
expression :: type
```

Arguments

expression

Expression qui correspond à une ou plusieurs valeurs, par exemple un nom de colonne ou un littéral. La conversion de valeurs null renvoie des valeurs null. L'expression ne peut pas contenir des chaînes vides.

type

L'un des Types de données pris en charge.

Type de retour

CAST renvoie le type de données spécifié par l'argument type.

> **Note**
>
> Amazon Redshift renvoie une erreur si vous essayez d'effectuer une conversion problématique, telle que la conversion DECIMAL suivante qui perd en précision :
>
> ```
> select 123.456::decimal(2,1);
> ```

ou une conversion INTEGER qui entraîne un dépassement de capacité :

```
select 12345678::smallint;
```

Exemples

Certains exemples utilisent l'exemple de base de données TICKIT. Pour plus d'informations sur la configuration d'échantillons de données, voir Charger des données.

Les deux requêtes suivantes sont équivalentes. Toutes deux convertissent une valeur décimale en un nombre entier :

```
select cast(pricepaid as integer)
from sales where salesid=100;

pricepaid
-----------
162
(1 row)
```

```
select pricepaid::integer
from sales where salesid=100;

pricepaid
-----------
162
(1 row)
```

Ce qui suit produit un résultat similaire. Il ne nécessite pas d'exemples de données pour s'exécuter :

```
select cast(162.00 as integer) as pricepaid;

pricepaid
-----------
162
(1 row)
```

Dans cet exemple, les valeurs d'une colonne d'horodatage sont converties en dates, ce qui entraîne la suppression de l'horodatage de chaque résultat :

```
select cast(saletime as date), salesid
from sales order by salesid limit 10;

 saletime  | salesid
-----------+---------
2008-02-18 |       1
2008-06-06 |       2
2008-06-06 |       3
2008-06-09 |       4
2008-08-31 |       5
2008-07-16 |       6
2008-06-26 |       7
2008-07-10 |       8
2008-07-22 |       9
2008-08-06 |      10
(10 rows)
```

Si vous n'avez pas utilisé CAST comme illustré dans l'exemple précédent, les résultats incluraient l'heure : 2008-02-18 02:36:48.

La requête suivante convertit des données de caractères variables en date. Elle ne nécessite pas d'exemples de données pour s'exécuter.

```
select cast('2008-02-18 02:36:48' as date) as mysaletime;

mysaletime
--------------------
2008-02-18
(1 row)
```

Dans cet exemple, les valeurs d'une colonne de dates sont converties en horodatages :

```
select cast(caldate as timestamp), dateid
from date order by dateid limit 10;

      caldate       | dateid
--------------------+--------
2008-01-01 00:00:00 |   1827
2008-01-02 00:00:00 |   1828
2008-01-03 00:00:00 |   1829
2008-01-04 00:00:00 |   1830
2008-01-05 00:00:00 |   1831
```

```
2008-01-06 00:00:00 |   1832
2008-01-07 00:00:00 |   1833
2008-01-08 00:00:00 |   1834
2008-01-09 00:00:00 |   1835
2008-01-10 00:00:00 |   1836
(10 rows)
```

Dans un cas comme dans l'exemple précédent, vous pouvez obtenir un contrôle supplémentaire sur le formatage de sortie en utilisant TO_CHAR.

Dans cet exemple, un nombre entier est converti en chaîne de caractères :

```
select cast(2008 as char(4));

bpchar
--------
2008
```

Dans cet exemple, une valeur DECIMAL(6,3) est convertie en valeur DECIMAL(4,1) :

```
select cast(109.652 as decimal(4,1));

numeric
---------
109.7
```

Cet exemple montre une expression plus complexe. La colonne PRICEPAID (colonne DECIMAL(8,2)) de la table SALES est convertie en une colonne DECIMAL(38,2) et les valeurs sont multipliées par 100000000000000000000 :

```
select salesid, pricepaid::decimal(38,2)*100000000000000000000
as value from sales where salesid<10 order by salesid;


 salesid |           value
---------+----------------------------
       1 | 728000000000000000000000.00
       2 |  760000000000000000000.00
       3 | 350000000000000000000000.00
       4 | 175000000000000000000000.00
       5 | 154000000000000000000000.00
       6 | 394000000000000000000000.00
```

```
       7 | 78800000000000000000000.00
       8 | 19700000000000000000000.00
       9 | 59100000000000000000000.00
(9 rows)
```

> **Note**
>
> Vous ne pouvez pas effectuer d'opération CAST ou CONVERT sur le type de données GEOMETRY pour le remplacer par un autre type de données. Toutefois, vous pouvez fournir une représentation hexadécimale du littéral d'une chaîne au format EWKB (extended well-known binary) en tant qu'entrée des fonctions qui acceptent un argument GEOMETRY. Par exemple, la fonction ST_AsText suivante attend un type de données GEOMETRY.
>
> ```
> SELECT ST_AsText('010100000000000000000001C40000000000002040');
> ```
>
> ```
> st_astext
> ------------
>  POINT(7 8)
> ```
>
> Vous pouvez également spécifier de façon explicite le type de données GEOMETRY.
>
> ```
> SELECT ST_AsText('010100000000000000000014400000000000001840'::geometry);
> ```
>
> ```
> st_astext
> ------------
>  POINT(5 6)
> ```

Fonction CONVERT

Comme la fonction CAST, la fonction CONVERT convertit un type de données en un autre type de données compatible. Par exemple, vous pouvez convertir une chaîne en date ou un type numérique en chaîne. CONVERT effectue une conversion au moment de l'exécution, ce qui signifie que la conversion ne modifie pas le type de données d'une valeur dans une table source. Elle n'est modifiée que dans le contexte de la requête.

Certains types de données nécessitent une conversion explicite en d'autres types de données à l'aide de la fonction CONVERT. D'autres types de données peuvent être convertis implicitement,

dans le cadre d'une autre commande, sans utiliser CAST ou CONVERT. Consultez Compatibilité et conversion de types.

Syntaxe

```
CONVERT ( type, expression )
```

Arguments

type

L'un des Types de données pris en charge.

expression

Expression qui correspond à une ou plusieurs valeurs, par exemple un nom de colonne ou un littéral. La conversion de valeurs null renvoie des valeurs null. L'expression ne peut pas contenir des chaînes vides.

Type de retour

CONVERT renvoie le type de données spécifié par l'argument type.

> **Note**
>
> Amazon Redshift renvoie une erreur si vous essayez d'effectuer une conversion problématique, telle que la conversion DECIMAL suivante qui perd en précision :
>
> ```
> SELECT CONVERT(decimal(2,1), 123.456);
> ```
>
> ou une conversion INTEGER qui entraîne un dépassement de capacité :
>
> ```
> SELECT CONVERT(smallint, 12345678);
> ```

Exemples

Certains exemples utilisent l'exemple de base de données TICKIT. Pour plus d'informations sur la configuration d'échantillons de données, voir Charger des données.

La requête suivante utilise la fonction CONVERT pour convertir une colonne de décimales en nombres entiers.

```
SELECT CONVERT(integer, pricepaid)
FROM sales WHERE salesid=100;
```

Cet exemple convertit un nombre entier en une chaîne de caractères.

```
SELECT CONVERT(char(4), 2008);
```

Dans cet exemple, la date et l'heure actuelles sont converties en un type de données de caractères variables :

```
SELECT CONVERT(VARCHAR(30), GETDATE());

getdate
---------
2023-02-02 04:31:16
```

Cet exemple convertit la colonne saletime en heure uniquement, en supprimant les dates de chaque ligne.

```
SELECT CONVERT(time, saletime), salesid
FROM sales order by salesid limit 10;
```

Pour plus d'informations sur la conversion d'un horodatage d'un fuseau horaire vers un autre, consultez Fonction CONVERT_TIMEZONE. Pour les autres fonctions de date et d'heure, consultez Fonctions de date et d'heure.

L'exemple suivant convertit des données à caractères variables en un objet de type datetime.

```
SELECT CONVERT(datetime, '2008-02-18 02:36:48') as mysaletime;
```

> **Note**
>
> Vous ne pouvez pas effectuer d'opération CAST ou CONVERT sur le type de données GEOMETRY pour le remplacer par un autre type de données. Toutefois, vous pouvez fournir une représentation hexadécimale du littéral d'une chaîne au format EWKB (extended well-

known binary) en tant qu'entrée des fonctions qui acceptent un argument GEOMETRY. Par exemple, la fonction ST_AsText suivante attend un type de données GEOMETRY.

```
SELECT ST_AsText('010100000000000000000001C40000000000002040');
```

```
st_astext
------------
 POINT(7 8)
```

Vous pouvez également spécifier de façon explicite le type de données GEOMETRY.

```
SELECT ST_AsText('010100000000000000000014400000000000001840'::geometry);
```

```
st_astext
------------
 POINT(5 6)
```

TEXT_TO_INT_ALT

TEXT_TO_INT_ALT convertit une chaîne de caractères en nombre entier à l'aide d'un formatage de style Teradata. Dans le résultat, les chiffres après la virgule sont tronqués.

Syntaxe

```
TEXT_TO_INT_ALT (expression [ , 'format'])
```

Arguments

expression

Expression qui génère une ou plusieurs valeurs CHAR ou VARCHAR, telles qu'un nom de colonne ou une chaîne littérale. La conversion de valeurs null renvoie des valeurs null. La fonction convertit les chaînes vides en 0.

format

Littéral de chaîne qui définit le format de l'expression en entrée. Pour plus d'informations sur les caractères de formatage que vous pouvez spécifier, consultez Caractères de formatage de type Teradata pour les données numériques.

Type de retour

TEXT_TO_INT_ALT renvoie une valeur INTEGER.

La partie après la virgule du résultat de la conversion est tronquée.

Amazon Redshift renvoie une erreur si la conversion dans la phrase format que vous spécifiez n'a pas réussi.

Exemples

L'exemple suivant convertit la chaîne d'expression en entrée 123- en nombre entier -123.

```
select text_to_int_alt('123-');
```

```
text_to_int_alt
----------
      -123
```

L'exemple suivant convertit la chaîne d'expression en entrée 2147483647+ en nombre entier 2147483647.

```
select text_to_int_alt('2147483647+');
```

```
text_to_int_alt
----------
2147483647
```

L'exemple suivant convertit la chaîne d'expression en entrée exponentielle -123E-2 en nombre entier -1.

```
select text_to_int_alt('-123E-2');
```

```
text_to_int_alt
----------
        -1
```

L'exemple suivant convertit la chaîne d'expression en entrée 2147483647+ en nombre entier 2147483647.

```
select text_to_int_alt('2147483647+');
```

```
text_to_int_alt
----------
2147483647
```

L'exemple suivant convertit la chaîne d'expression en entrée 123{ avec la phrase format 999S en nombre entier 1230. Le caractère S indique une « Signed Zoned Decimal ». Pour de plus amples informations, veuillez consulter Caractères de formatage de type Teradata pour les données numériques.

```
select text_to_int_alt('123{', '999S');
```

```
text_to_int_alt
----------
      1230
```

L'exemple suivant convertit la chaîne d'expression d'entrée « USD123 » avec la phrase de format « C9 (I) » en un entier 123. Consultez Caractères de formatage de type Teradata pour les données numériques.

```
select text_to_int_alt('USD123', 'C9(I)');
```

```
text_to_int_alt
----------
       123
```

L'exemple suivant spécifie une colonne de table comme étant l'expression en entrée.

```
select text_to_int_alt(a), text_to_int_alt(b) from t_text2int order by 1;
```

```
 text_to_int_alt | text_to_int_alt
-----------------+-----------------
            -123 |            -123
            -123 |            -123
             123 |             123
             123 |             123
```

Voici la définition de la table et l'instruction insert pour cet exemple.

```
create table t_text2int (a varchar(200), b char(200));
```

```
insert into t_text2int VALUES('123', '123'),('123.123', '123.123'), ('-123', '-123'),
 ('123-', '123-');
```

TEXT_TO_NUMERIC_ALT

TEXT_TO_NUMERIC_ALT effectue une opération de conversion de style Teradata pour convertir une chaîne de caractères en un format de données numériques.

Syntaxe

```
TEXT_TO_NUMERIC_ALT (expression [, 'format'] [, precision, scale])
```

Arguments

expression

Expression qui correspond à une ou plusieurs valeurs CHAR ou VARCHAR, par exemple, un nom de colonne ou un littéral. La conversion de valeurs null renvoie des valeurs null. Les chaînes vides sont converties en 0.

format

Littéral de chaîne qui définit le format de l'expression en entrée. Pour plus d'informations, consultez Caractères de formatage de type Teradata pour les données numériques.

precision

Nombre de chiffres dans le résultat numérique. La valeur par défaut est 38.

échelle

Nombre de chiffres à droite de la virgule décimale dans le résultat numérique. La valeur par défaut est 0.

Type de retour

TEXT_TO_NUMERIC_ALT renvoie un nombre DECIMAL.

Amazon Redshift renvoie une erreur si la conversion dans la phrase format que vous spécifiez n'a pas réussi.

Amazon Redshift convertit la chaîne d'expression en entrée en type numérique avec la précision la plus élevée que vous spécifiez pour ce type dans l'option precision. Si la longueur de la valeur numérique dépasse la valeur que vous spécifiez pour precision, Amazon Redshift arrondit la valeur numérique en appliquant les règles suivantes :

- Si la longueur du résultat de la conversion dépasse la longueur que vous spécifiez dans la phrase format, Amazon Redshift renvoie une erreur.
- Si le résultat est converti en une valeur numérique, le résultat est arrondi à la valeur la plus proche. Si la partie après la virgule se trouve exactement à mi-chemin entre les résultats supérieur et inférieur de la conversion, le résultat est arrondi à la valeur pair la plus proche.

Exemples

L'exemple suivant convertit la chaîne d'expression en entrée 1.5 en valeur numérique 2. Parce que l'instruction ne spécifie pas d'échelle, l'échelle par défaut est 0 et le résultat de la conversion n'inclut pas les chiffres après la virgule. Etant donné que .5 est à mi-chemin entre 1 et 2, le résultat de la conversion est arrondi à la valeur pair de 2.

```
select text_to_numeric_alt('1.5');
```

```
 text_to_numeric_alt
---------------------
                   2
```

L'exemple suivant convertit la chaîne d'expression en entrée 2.51 en valeur numérique 3. Parce que l'instruction ne spécifie pas d'échelle, la valeur d'échelle par défaut est 0 et le résultat de la conversion n'inclut pas les chiffres après la virgule. Étant donné que .51 est plus proche de 3 que 2, le résultat de la conversion est arrondi à la valeur de 3.

```
select text_to_numeric_alt('2.51');
```

```
 text_to_numeric_alt
---------------------
                   3
```

L'exemple suivant convertit la chaîne d'expression en entrée 123.52501 avec une précision de 10 et une échelle de 2 en valeur numérique 123,53.

```
select text_to_numeric_alt('123.52501', 10, 2);
```

```
 text_to_numeric_alt
---------------------
              123.53
```

L'exemple suivant convertit la chaîne d'expression d'entrée « 123 {» avec la phrase de format « 999S » en 1230 numérique. Le caractère S indique une « Signed Zoned Decimal ». Pour de plus amples informations, veuillez consulter Caractères de formatage de type Teradata pour les données numériques.

```
select text_to_numeric_alt('123{', '999S');
```

```
text_to_int_alt
----------
      1230
```

L'exemple suivant convertit la chaîne d'expression d'entrée « USD123 » avec la phrase de format « C9 (I) » en 124 numérique. Consultez Caractères de formatage de type Teradata pour les données numériques.

```
select text_to_numeric_alt('USD123.9', 'C9(I)');
```

```
text_to_numeric_alt
----------
       124
```

L'exemple suivant spécifie une colonne de table comme étant l'expression en entrée.

```
select text_to_numeric_alt(a), text_to_numeric_alt(b) from t_text2numeric order by 1;
```

```
          text_to_numeric_alt           |          text_to_numeric_alt
----------------------------------------+----------------------------------------
 -99999999999999999999999999999999999999 | -99999999999999999999999999999999999999
                                  -12300 |                                  -12300
```

```
                                    123 |                                    123
                                    123 |                                    123
 99999999999999999999999999999999999999 |  99999999999999999999999999999999999999
```

Voici la définition de la table et l'instruction insert pour cet exemple.

```
create table  t_text2numeric (a varchar(200), b char(200));
```

```
insert into  t_text2numeric values
('123', '123'),
('+123.456', '+123.456'),
('-' || repeat('9', 38), '-' || repeat('9', 38)),
(repeat('9', 38) || '+', repeat('9', 38) || '+'),
('-123E2', '-123E2');
```

TO_CHAR

TO_CHAR convertit un horodatage ou une expression numérique en un format de données de chaînes de caractères.

Syntaxe

```
TO_CHAR (timestamp_expression | numeric_expression , 'format')
```

Arguments

timestamp_expression

Expression qui se traduit par une valeur de type TIMESTAMP ou TIMESTAMPTZ ou par une valeur qui peut être implicitement convertie en un horodatage.

numeric_expression

Expression qui se traduit par une valeur de type de données numérique ou par une valeur qui peut être implicitement convertie en un type numérique. Pour plus d'informations, consultez Types numériques. TO_CHAR insère un espace à gauche de la chaîne numérique.

> **Note**
>
> TO_CHAR ne prend pas en charge les valeurs DECIMAL de 128 bits.

format

Format de la nouvelle valeur. Pour connaître les formats valides, consultez Chaînes de format datetime et Chaînes de format numériques.

Type de retour

VARCHAR

Exemples

L'exemple suivant convertit un horodatage en une valeur contenant la date et l'heure dans un format comprenant le nom du mois rempli jusqu'à neuf caractères, le nom du jour de la semaine et le numéro du jour du mois.

```
select to_char(timestamp '2009-12-31 23:15:59', 'MONTH-DY-DD-YYYY HH12:MIPM');

to_char
-------------------------
DECEMBER -THU-31-2009 11:15PM
```

L'exemple suivant convertit un horodatage en une valeur avec le numéro du jour de l'année.

```
select to_char(timestamp '2009-12-31 23:15:59', 'DDD');

to_char
-------------------------
365
```

L'exemple suivant convertit un horodatage en un numéro de jour de la semaine ISO.

```
select to_char(timestamp '2022-05-16 23:15:59', 'ID');

to_char
-------------------------
1
```

L'exemple suivant extrait le nom du mois d'une date.

```
select to_char(date '2009-12-31', 'MONTH');
```

```
to_char
------------------------
DECEMBER
```

L'exemple suivant convertit chaque valeur STARTTIME de la table EVENT en une chaîne composée d'heures, de minutes et de secondes.

```
select to_char(starttime, 'HH12:MI:SS')
from event where eventid between 1 and 5
order by eventid;

to_char
----------
02:30:00
08:00:00
02:30:00
02:30:00
07:00:00
```

L'exemple suivant convertit une valeur d'horodatage complète en un format différent.

```
select starttime, to_char(starttime, 'MON-DD-YYYY HH12:MIPM')
from event where eventid=1;

      starttime      |       to_char
---------------------+---------------------
 2008-01-25 14:30:00 | JAN-25-2008 02:30PM
```

L'exemple suivant convertit un littéral d'horodatage en une chaîne de caractères.

```
select to_char(timestamp '2009-12-31 23:15:59','HH24:MI:SS');

to_char
----------
23:15:59
```

L'exemple suivant convertit un nombre décimal en chaîne de caractères.

```
select to_char(125.8, '999.99');

to_char
```

```
---------
125.80
```

L'exemple suivant convertit un nombre décimal en chaîne de caractères.

```
select to_char(125.8, '999D99');

to_char
---------
125.80
```

L'exemple suivant convertit un nombre en chaîne de caractères commençant par un zéro.

```
select to_char(125.8, '0999D99');

to_char
---------
0125.80
```

L'exemple suivant convertit un nombre en chaîne de caractères avec le signe moins à la fin.

```
select to_char(-125.8, '999D99S');

to_char
---------
125.80-
```

L'exemple suivant convertit un nombre en chaîne de caractères avec le signe positif ou négatif à la position spécifiée.

```
select to_char(125.8, '999D99SG');

to_char
---------
125.80+
```

L'exemple suivant convertit un nombre en chaîne de caractères avec le signe positif à la position spécifiée.

```
select to_char(125.8, 'PL999D99');
```

```
to_char
---------
+ 125.80
```

L’exemple suivant convertit un nombre en chaîne de caractères avec le symbole de la devise.

```
select to_char(-125.88, '$S999D99');

to_char
---------
$-125.88
```

L'exemple suivant convertit un nombre en chaîne de caractères avec le symbole monétaire à la position spécifiée.

```
select to_char(-125.88, 'S999D99L');

to_char
---------
-125.88$
```

L'exemple suivant convertit un nombre en chaîne de caractères à l'aide d'un séparateur de milliers (virgule).

```
select to_char(1125.8, '9,999.99');

to_char
---------
1,125.80
```

L’exemple suivant convertit un nombre en une chaîne de caractères.

```
select to_char(-125.88, '$999D99PR');

to_char
---------
$<125.88>
```

L’exemple suivant convertit un nombre en chaîne numérale romaine.

```
select to_char(125, 'RN');

to_char
---------
   CXXV
```

L'exemple suivant convertit une date en code du siècle.

```
select to_char(date '2020-12-31', 'CC');

to_char
---------
21
```

L'exemple suivant affiche le jour de la semaine.

```
SELECT to_char(current_timestamp, 'FMDay, FMDD HH12:MI:SS');

to_char
-----------------------
Wednesday, 31 09:34:26
```

L'exemple suivant affiche le suffixe ordinal d'un nombre.

```
SELECT to_char(482, '999th');

to_char
-----------------------
 482nd
```

L'exemple suivant soustraie la commission du prix d'achat de la table des ventes. La différence est ensuite arrondie et convertie en un chiffre romain, indiqué dans la colonne to_char :

```
select salesid, pricepaid, commission, (pricepaid - commission)
as difference, to_char(pricepaid - commission, 'rn') from sales
group by sales.pricepaid, sales.commission, salesid
order by salesid limit 10;

 salesid | pricepaid | commission | difference |     to_char
---------+-----------+------------+------------+-----------------
       1 |    728.00 |     109.20 |     618.80 |           dcxix
```

```
       2 |     76.00 |      11.40 |      64.60 |            lxv
       3 |    350.00 |      52.50 |     297.50 |       ccxcviii
       4 |    175.00 |      26.25 |     148.75 |          cxlix
       5 |    154.00 |      23.10 |     130.90 |          cxxxi
       6 |    394.00 |      59.10 |     334.90 |        cccxxxv
       7 |    788.00 |     118.20 |     669.80 |          dclxx
       8 |    197.00 |      29.55 |     167.45 |         clxvii
       9 |    591.00 |      88.65 |     502.35 |            dii
      10 |     65.00 |       9.75 |      55.25 |             lv
```

L'exemple suivant ajoute le symbole de la devise aux valeurs de différence affichées dans la colonne to_char :

```
select salesid, pricepaid, commission, (pricepaid - commission)
as difference, to_char(pricepaid - commission, 'l99999D99') from sales
group by sales.pricepaid, sales.commission, salesid
order by salesid limit 10;

salesid | pricepaid | commission | difference |  to_char
--------+-----------+------------+------------+------------
      1 |    728.00 |     109.20 |     618.80 | $   618.80
      2 |     76.00 |      11.40 |      64.60 | $    64.60
      3 |    350.00 |      52.50 |     297.50 | $   297.50
      4 |    175.00 |      26.25 |     148.75 | $   148.75
      5 |    154.00 |      23.10 |     130.90 | $   130.90
      6 |    394.00 |      59.10 |     334.90 | $   334.90
      7 |    788.00 |     118.20 |     669.80 | $   669.80
      8 |    197.00 |      29.55 |     167.45 | $   167.45
      9 |    591.00 |      88.65 |     502.35 | $   502.35
     10 |     65.00 |       9.75 |      55.25 | $    55.25
```

L'exemple suivant répertorie le siècle au cours duquel chaque vente a été effectuée.

```
select salesid, saletime, to_char(saletime, 'cc') from sales
order by salesid limit 10;

 salesid |      saletime       | to_char
---------+---------------------+---------
       1 | 2008-02-18 02:36:48 | 21
       2 | 2008-06-06 05:00:16 | 21
       3 | 2008-06-06 08:26:17 | 21
       4 | 2008-06-09 08:38:52 | 21
       5 | 2008-08-31 09:17:02 | 21
```

```
       6 | 2008-07-16 11:59:24 | 21
       7 | 2008-06-26 12:56:06 | 21
       8 | 2008-07-10 02:12:36 | 21
       9 | 2008-07-22 02:23:17 | 21
      10 | 2008-08-06 02:51:55 | 21
```

L'exemple suivant convertit chaque valeur STARTTIME de la table EVENT en une chaîne qui se compose d'heures, de minutes, de secondes et d'un fuseau horaire.

```
select to_char(starttime, 'HH12:MI:SS TZ')
from event where eventid between 1 and 5
order by eventid;

to_char
----------
02:30:00 UTC
08:00:00 UTC
02:30:00 UTC
02:30:00 UTC
07:00:00 UTC
```

L'exemple suivant illustre la mise en forme des secondes, millisecondes et microsecondes.

```
select sysdate,
to_char(sysdate, 'HH24:MI:SS') as seconds,
to_char(sysdate, 'HH24:MI:SS.MS') as milliseconds,
to_char(sysdate, 'HH24:MI:SS:US') as microseconds;

timestamp           | seconds  | milliseconds | microseconds
--------------------+----------+--------------+----------------
2015-04-10 18:45:09 | 18:45:09 | 18:45:09.325 | 18:45:09:325143
```

Fonction TO_DATE

TO_DATE convertit une date représentée par une chaîne de caractères en un type de données DATE.

> **Note**
>
> TO_DATE ne prend pas en charge les chaînes de format avec Q (numéro de trimestre).

Syntaxe

```
TO_DATE(string, format)
```

```
TO_DATE(string, format, is_strict)
```

Arguments

string

Chaîne à convertir.

format

Littéral de chaîne qui définit le format de la chaîne de sortie, en fonction de ses parties de date. Pour obtenir la liste des formats de jour, de mois et d'année valides, consultez Chaînes de format datetime.

is_strict

Valeur booléenne facultative qui spécifie si une erreur est renvoyée lorsqu'une valeur de date d'entrée est hors plage. Quand is_strict est défini sur TRUE, une erreur est renvoyée s'il y a une valeur hors plage. Quand is_strict est défini sur FALSE, qui est la valeur par défaut, les valeurs en dépassement sont acceptées.

Type de retour

TO_DATE renvoie une DATE, selon la valeur de format.

Si la conversion au format échoue, une erreur est renvoyée.

Exemples

L'instruction SQL suivante convertit la date `02 Oct 2001` en type de données de date.

```
select to_date('02 Oct 2001', 'DD Mon YYYY');

to_date
------------
2001-10-02
(1 row)
```

L'instruction SQL suivante convertit la chaîne 20010631 en date.

```
select to_date('20010631', 'YYYYMMDD', FALSE);
```

Le résultat est le 1er juillet 2001, car il n'y a que 30 jours en juin.

```
to_date
------------
2001-07-01
```

L'instruction SQL suivante convertit la chaîne 20010631 en date :

```
to_date('20010631', 'YYYYMMDD', TRUE);
```

Le résultat est une erreur car il n'y a que 30 jours en juin.

```
ERROR:  date/time field date value out of range: 2001-6-31
```

TO_NUMBER

TO_NUMBER convertit une chaîne en une valeur numérique (décimale).

> **Note**
>
> Nous vous recommandons d'utiliser FM dans votre chaîne de format pour supprimer les blancs et les zéros de remplissage. Pour obtenir la liste des formats valides, consultez Chaînes de format numériques.

Syntaxe

```
to_number(string, format)
```

Arguments

string

Chaîne à convertir. Le format doit être une valeur littérale.

format

Le deuxième argument est une chaîne de format qui indique comment la chaîne de caractères doit être analysée afin de créer la valeur numérique. Par exemple, le format `'FM99D999'` spécifie que la chaîne à convertir se compose de cinq chiffres, avec la virgule à la troisième position. Par exemple, `to_number('12.345','FM99D999')` renvoie `12.345` comme une valeur numérique. Pour obtenir la liste des formats valides, consultez Chaînes de format numériques.

Type de retour

TO_NUMBER renvoie un nombre DECIMAL.

Si la conversion au format échoue, une erreur est renvoyée.

Exemples

L'exemple suivant convertit la chaîne `12,454.8-` en un nombre :

```
select to_number('12,454.8-', 'FM99G999D9S');

to_number
-----------
-12454.8
```

L'exemple suivant convertit la chaîne `$ 12,454.88` en un nombre :

```
select to_number('$ 12,454.88', 'FML99G999D99');

to_number
-----------
12454.88
```

L'exemple suivant convertit la chaîne `$ 2,012,454.88` en un nombre :

```
select to_number('$ 2,012,454.88', 'FML9,999,999.99');

to_number
-----------
2012454.88
```

Fonction TRY_CAST

Par rapport à la fonction CAST, TRY_CAST tente d'abord de convertir l'expression dans le type spécifié. Si le casting échoue en raison d'erreurs de conversion, l'opération renvoie la valeur null. Si une conversion n'est pas explicitement autorisée, l'opération renvoie une erreur. Vous trouverez la liste des conversions possibles dans les notes d'utilisation ci-dessous. Par exemple, la tentative de conversion d'un booléen en horodatage n'est pas autorisée.

Syntaxe

```
TRY_CAST(expression AS type)
```

Arguments

expression

Expression qui correspond à une ou plusieurs valeurs, par exemple un nom de colonne ou un littéral. La conversion de valeurs null renvoie des valeurs null. L'expression ne peut pas contenir des chaînes vides.

type

L'un des types de données pris en charge. Pour obtenir la liste complète des types de données, consultezTypes de données. Pour obtenir la liste des paires de types de données source et de type de données cibles pris en charge, consultez les notes d'utilisation ci-dessous.

Type de retour

TRY_CAST renvoie une valeur du type de données spécifié par l'argument type. Si la conversion échoue, l'opération renvoie la valeur null.

Notes d'utilisation

Vous trouverez ci-dessous la liste des paires de types de données source et de type de données cible prises en charge par Amazon Redshift pour TRY_CAST.

BOOL

SMALLINT, INT, BIGINT, SUPER

PETITE MENTHE

BOOL, INT, BIGINT, DECIMAL, REAL, FLOAT, BPCHAR, TEXT, VARCHAR, SUPER

INT

BOOL, SMALLINT, BIGINT, DECIMAL, REAL, FLOAT, BOCHAR, TEXT, VARCHAR, SUPER

BIGINT

BOOL, SMALLINT, INT, DECIMAL, REAL, FLOAT, BOCHAR, TEXT, VARCHAR, SUPER

DÉCIMAL

SMALLINT, INT, BIGINT, REAL, FLOAT, BOCHAR, TEXT, VARCHAR, SUPER

RÉEL

SMALLINT, INT, BIGINT, DECIMAL, FLOAT, BOCHAR, TEXT, VARCHAR, SUPER

FLOTTEUR

SMALLINT, INT, BIGINT, DECIMAL, REAL, BOCHAR, TEXT, VARCHAR, SUPER

BPCHAR

SMALLINT, INT, BIGINT, DECIMAL, REAL, FLOAT, TEXT, VARCHAR, TIMESTAMP, DATE, HEURE, TIMETZ, SUPER

TEXTE

SMALLINT, INT, BIGINT, DECIMAL, REAL, FLOAT, BOCHAR, VARCHAR, TIMESTAMP, DATE, HEURE, HEURES, SUPER

VARCHAR

SMALLINT, INT, BIGINT, DÉCIMAL, RÉEL, FLOAT, BOCHAR, TEXTE, HORODATAGE, HORODATAGE, DATE, HEURE, HEURES, SUPER

HORODATAGE

BPCHAR, TEXTE, VARCHAR, HORODATAGE, DATE, HEURE

HORODATAGE TZ

BPCHAR, TEXTE, VARCHAR, HORODATAGE, DATE, HEURE, HEURES

DATE

BPCHAR, TEXTE, VARCHAR, HORODATAGE, TIMESTAMPTZ

HEURE

BPCHAR, TEXTE, VARCHAR

TIMETZ

BPCHAR, TEXTE, VARCHAR

SUPER

SUPER peut être converti en n'importe quel autre type de données, à l'exception de DATE, TIME, TIMETZ, TIMESTAMP et TIMESTAMPTZ.

Exemples

L'exemple suivant transforme une STRING en INTEGER.

```
SELECT TRY_CAST('123' AS INT);

int
----
123
```

L'exemple suivant renvoie la valeur null. La conversion d'une STRING en INTEGER est autorisée afin que TRY_CAST ne renvoie pas d'erreur, mais « foo » n'est pas un entier, donc la fonction renvoie null.

```
SELECT TRY_CAST('foo' AS INT)
```

L'exemple suivant renvoie une erreur, car la conversion d'un BOOLEAN en TIMESTAMP n'est pas autorisée.

```
SELECT TRY_CAST(true as timestamp);
```

Comme TRY_CAST renvoie la valeur null au lieu de renvoyer immédiatement une erreur en cas d'échec de la conversion, vous pouvez utiliser TRY_CAST pour filtrer les données non valides. Prenons l'exemple suivant, où une ligne non valide est filtrée en raison d'un échec de conversion dans la colonne d'âge d'Akua Mansa.

```
CREATE TABLE IF NOT EXISTS student_data(
name VARCHAR(100) NOT NULL,
age VARCHAR(3) NOT NULL,
enrollment_date DATE NOT NULL);
```

```
INSERT INTO student_data (name, age, enrollment_date)
VALUES
('Alejandro Rosalez', '10', '01/01/2000'),
('Akua Mansa', 'Ten', '01/01/2000');

SELECT * FROM student_data WHERE TRY_CAST(age AS INT) IS NOT NULL;

--Akua is not returned.
 name              | age | enrollment_date
-------------------+-----+-----------------
 Alejandro Rosalez | 10  | 01/01/2000
```

Chaînes de format datetime

Vous pouvez trouver une référence pour les chaînes de format datetime ci-après.

Les chaînes de format suivantes s'appliquent à des fonctions telles que TO_CHAR. Ces chaînes peuvent contenir des séparateurs datetime (comme '-', '/', ou ':') et les « dateparts » et « timeparts » suivantes.

Partie de date ou d'horodatage	Signification
BC ou B.C., AD ou A.D., b.c. ou bc, ad ou a.d.	Indicateurs d'ère en majuscules et en minuscules
CC	Nombre du siècle à deux chiffres
YYYY, YYY, YY, Y	Nombre de l'année à 4 chiffres, 3 chiffres, 2 chiffres, 1 chiffre
Y,YYY	Numéro de l'année à 4 chiffres avec virgule
IYYY, IYY, IY, I	Numéro de l'année ISO (International Organization for Standardization) 4 chiffres, 3 chiffres, 2 chiffres, 1 chiffre
Q	Numéro de trimestre (1 à 4)
MONTH, Month, month	Nom du mois (majuscules, à casse mixte, minuscules, 9 caractères avec ajout de blancs)

Partie de date ou d'horodatage	Signification
MON, Mon, mon	Nom du mois abrégé (majuscules, à casse mixte, minuscules, trois caractères avec ajout de blancs)
MM	Numéro du mois (01-12)
RM, rm	Numéro du mois en chiffres romains (de I à XII, I représentant janvier, en majuscules ou minuscules)
W	Semaine du mois (de 1 à 5, la première semaine commence le premier jour du mois.)
WW	Numéro de la semaine de l'année (de 1 à 53, la première semaine commence le premier jour de l'année.)
IW	Numéro de semaine ISO de l'année (le premier jeudi de la nouvelle année se trouve dans la semaine 1.)
DAY, Day, day	Nom du jour (majuscules, à casse mixte, minuscules, 9 caractères avec ajout de blancs)
DY, Dy, dy	Nom du jour abrégé (majuscules, à casse mixte, minuscules, 3 caractères avec ajout de blancs)
DDD	Jour de l'année (de 001 à 366)
IDDD	Jour de la semaine de l'année de numérotation ISO 8601 (001-371 ; le premier jour de l'année est le lundi de la première semaine ISO)
DD	Jour du mois en tant que nombre (de 01 à 31)

Partie de date ou d'horodatage	Signification
D	Jour de la semaine (de 1 à 7 ; le dimanche étant le 1) **Note** La partie de date D se comporte différemment de la partie de date jour de la semaine (DOW) utilisée pour les fonctions datetime DATE_PART et EXTRACT. DOW s'appuie sur des nombres entiers compris entre 0 et 6, dimanche étant le 0. Pour plus d'informations, consultez Parties de date pour les fonctions de date ou d'horodatage.
ID	ISO 8601, le jour de la semaine, du lundi (1) au dimanche (7)
J	Jour julien (jours depuis le 1er janvier 4712 av. J.-C.)
HH24	Heure (24 heures, de 00 à 23)
HH ou HH12	Heure (12 heures, de 01 à 12)
MI	Minutes (00–59)
SS	Secondes (00–59)
MS	Millisecondes (0,000)
US	Microsecondes (0,000000)
AM ou PM, A.M. ou P.M., a.m. ou p.m., am ou pm	Indicateurs des méridiens en majuscules et en minuscules (pour 12 heures)

Partie de date ou d'horodatage	Signification
TZ, tz	Abréviation de fuseau horaire en majuscules et minuscules ; valide pour TIMESTAMPTZ uniquement
OF	Décalage de l'UTC ; valide pour TIMESTAMPTZ uniquement

> **Note**
>
> Vous devez placer les séparateurs d'heure et de date (tels que '-', '/' ou ':') entre guillemets simples, mais vous devez placer les éléments "dateparts" et "timeparts" figurant dans la table précédente entre guillemets doubles (").

Exemples

Pour des exemples de formatage de dates sous forme de chaînes, consultez TO_CHAR.

Chaînes de format numériques

Vous trouverez ci-dessous une référence pour les chaînes de format numérique.

Les chaînes de format suivantes s'appliquent aux fonctions telles que TO_NUMBER et TO_CHAR.

- Pour des exemples de formatage de chaînes sous forme de nombres, consultez TO_NUMBER.
- Pour des exemples de formatage de nombres sous forme de chaînes, consultez TO_CHAR.

Format	Description
9	Valeur numérique avec le nombre spécifié de chiffres.
0	Valeur numérique commençant par des zéros.
. (point), D	Virgule.

Format	Description
, (comma)	Séparateur de milliers.
CC	Code de siècle. Par exemple, le 21e siècle a commencé le 2001-01-01 (pris en charge pour TO_CHAR uniquement).
FM	Mode de remplissage. Permet de supprimer les zéros et les vides de remplissage.
PR	Valeur négative entre crochets.
S	Signe fixé à un nombre.
L	Symbole de devise à la position spécifiée.
G	Séparateur de groupe.
MI	Signe moins à la position spécifiée pour les numéros inférieurs à 0.
PL	Signe plus à la position spécifiée pour les numéros supérieurs à 0.
SG	Signe plus ou moins à la position spécifiée.
RN	Chiffre romains compris entre 1 et 3 999 (pris en charge pour TO_CHAR uniquement).
TH ou th	Suffixe de nombre ordinal. Ne convertit pas les nombres ou les valeurs fractionnaires inférieurs à zéro.

Caractères de formatage de type Teradata pour les données numériques

Vous trouverez ci-dessous comment les fonctions TEXT_TO_INT_ALT et TEXT_TO_NUMERIC_ALT interprètent les caractères dans la chaîne d'expression en entrée. Vous pouvez également retrouver la liste des caractères que vous pouvez spécifier dans la phrase format. En outre, vous trouverez une

description des différences entre le formatage de style Teradata et Amazon Redshift pour l'option format.

Format	Description
G	Non pris en charge en tant que séparateur de groupe dans la chaîne d'expression en entrée. Vous ne pouvez pas spécifier ce caractère dans la phrase format.
D	Symbole Radix. Vous pouvez spécifier ce caractère dans la phrase format. Ce caractère équivaut au . (point). Le symbole Radix ne peut pas apparaître dans une phrase format contenant l'un des caractère s suivants : • . (point) • S (« s » majuscule) • V (« v » majuscule)
/ , : %	Caractères d'insertion / (barre oblique), virgule (,), : (deux-points) et % (signe pourcentage). Vous ne pouvez pas inclure ces caractères dans la phrase format. Amazon Redshift ignore ces caractères dans la chaîne d'expression en entrée.
.	Période en tant que caractère radix, c'est-à-dire un point décimal. Ce caractère ne peut pas apparaître dans une phrase format contenant l'un des caractères suivants : • D (« d » majuscule)

Format	Description
	• S (« s » majuscule) • V (« v » majuscule)
B	Vous ne pouvez pas inclure le caractère espace (B) dans la phrase format. Dans la chaîne d'expression en entrée, les espaces de début et de fin sont ignorés et les espaces entre les chiffres ne sont pas autorisés.
+ -	Vous ne pouvez pas inclure de signe plus (+) ou moins (-) dans la phrase format. Cependant , le signe plus (+) et le signe moins (-) sont analysés implicitement en tant que partie de la valeur numérique s'ils apparaissent dans la chaîne d'expression en entrée.
V	Indicateur de position décimale. Ce caractère ne peut pas apparaître dans une phrase format contenant l'un des caractères suivants : • D (« d » majuscule) • . (point)
Z	Chiffre décimal avec zéro supprimé. Amazon Redshift tronque les zéros du début. Le caractère Z ne peut pas suivre un caractère 9. Le caractère Z doit se trouver à gauche du caractère radix si la partie après la virgule contient le caractère 9.
9	Chiffre décimal.

Format	Description
CHAR(n)	Pour ce format, vous pouvez spécifier les valeurs suivantes : • CHAR se compose des caractères Z ou 9. Amazon Redshift ne prend pas en charge un + (plus) ou un - (moins) dans la valeur CHAR. • n est une constante entière, I ou F. Pour I, il s'agit du nombre de caractères nécessaires pour afficher la partie avant la virgule des données numériques ou entières. Pour F, il s'agit du nombre de caractères nécessaires pour afficher la partie après la virgule des données numériques.
-	Caractère tiret (-). Vous ne pouvez pas inclure ce caractère dans la phrase format. Amazon Redshift ignore ce caractère dans la chaîne d'expression en entrée.

Format	Description
S	Signed Zoned Decimal. Le caractère S doit suivre le dernier chiffre décimal de la phrase format. Le dernier caractère de la chaîne d'expression en entrée et la conversion numérique correspondante sont répertoriés dans Caractères de formatage des données pour le formatage de données numériques de type Signed Zone Decimal, Teradata . Le caractère S ne peut pas apparaître dans une phrase format contenant l'un des caractère s suivants : • + (signe plus) • . (point) • D (« d » majuscule) • Z (« z » majuscule) • F (« f » majuscule) • E (« e » majuscule)
E	Notation exponentielle. La chaîne d'expressio n en entrée peut inclure le caractère d'exposan t. Vous ne pouvez pas spécifier E comme caractère d'exposant dans la phrase format.
FN9	Non pris en charge dans Amazon Redshift.
FNE	Non pris en charge dans Amazon Redshift.

Format	Description
$, USD, US Dollars	Signe dollar ($), symbole monétaire ISO (USD) et nom de devise « US Dollars ». Le symbole monétaire ISO « USD » et le nom de devise « US Dollars » sont sensibles à la casse. Amazon Redshift prend en charge uniquement la devise USD. La chaîne d'expression en entrée peut inclure des espaces entre le symbole monétaire USD et la valeur numérique, par exemple $ 123E2 ou 123E2 $.
L	Symbole monétaire. Ce caractère de symbole monétaire ne peut apparaître qu'une seule fois dans la phrase format. Vous ne pouvez pas spécifier de caractères de symbole monétaire répétés.
C	Symbole monétaire ISO. Ce caractère de symbole monétaire ne peut apparaître qu'une seule fois dans la phrase format. Vous ne pouvez pas spécifier de caractères de symbole monétaire répétés.
N	Nom complet de la devise. Ce caractère de symbole monétaire ne peut apparaître qu'une seule fois dans la phrase format. Vous ne pouvez pas spécifier de caractères de symbole monétaire répétés.
O	Symbole monétaire double. Vous ne pouvez pas spécifier ce caractère dans la phrase format.

Format	Description
U	Symbole monétaire ISO double. Vous ne pouvez pas spécifier ce caractère dans la phrase format.
A	Nom complet de la devise double. Vous ne pouvez pas spécifier ce caractère dans la phrase format.

Caractères de formatage des données pour le formatage de données numériques de type Signed Zone Decimal, Teradata

Vous pouvez utiliser les caractères suivants dans la phrase format des fonctions TEXT_TO_INT_ALT et TEXT_TO_NUMERIC_ALT pour une valeur Signed Zone Decimal.

Dernier caractère de la chaîne en entrée	Conversion numérique
{ ou 0	n... 0
A ou 1	n... 1
B ou 2	n... 2
C ou 3	n... 3
D ou 4	n... 4
E ou 5	n... 5
F ou 6	n... 6
G ou 7	n... 7
H ou 8	n... 8
I ou 9	n... 9
}	-n... 0

Dernier caractère de la chaîne en entrée	Conversion numérique
J	-n … 1
K	-n... 2
L	-n... 3
M	-n... 4
N	-n... 5
O	-n... 6
P	-n... 7
Q	-n... 8
R	-n... 9

Fonctions de date et d’heure

Dans cette section, vous trouverez des informations sur les fonctions scalaires de date et d’heure prises en charge par Amazon Redshift.

Rubriques

- Résumés des fonctions de date et d’heure
- Fonctions date et heure dans les transactions
- Fonctions de nœud principal uniquement obsolètes
- + Opérateur (concaténation)
- Fonction ADD_MONTHS
- Fonction AT TIME ZONE
- Fonction CONVERT_TIMEZONE
- Fonction CURRENT_DATE
- Fonction DATE_CMP
- Fonction DATE_CMP_TIMESTAMP

- Fonction DATE_CMP_TIMESTAMPTZ
- Fonction DATEADD
- Fonction DATEDIFF
- Fonction DATE_PART
- Fonction DATE_PART_YEAR
- Fonction DATE_TRUNC
- Fonction EXTRACT
- Fonction GETDATE
- Fonction INTERVAL_CMP
- Fonction LAST_DAY
- Fonction MONTHS_BETWEEN
- Fonction NEXT_DAY
- Fonction SYSDATE
- Fonction TIMEOFDAY
- Fonction TIMESTAMP_CMP
- Fonction TIMESTAMP_CMP_DATE
- Fonction TIMESTAMP_CMP_TIMESTAMPTZ
- Fonction TIMESTAMPTZ_CMP
- Fonction TIMESTAMPTZ_CMP_DATE
- Fonction TIMESTAMPTZ_CMP_TIMESTAMP
- Fonction TIMEZONE
- Fonction TO_TIMESTAMP
- Fonction TRUNC
- Parties de date pour les fonctions de date ou d'horodatage

Résumés des fonctions de date et d'heure

Fonction	Syntaxe	Renvoie
+ Opérateur (concaténation)	date + time	`TIMESTAMP` ou

Fonction	Syntaxe	Renvoie
Concatène une date à une heure de chaque côté du symbole + et renvoie une valeur TIMESTAMP ou TIMESTAMPTZ.		TIMESTAMPZ
ADD_MONTHS Ajoute le nombre de mois spécifié à un horodatage.	ADD_MONTHS ({date|timestamp}, integer)	TIMESTAMP
AT TIME ZONE Spécifie le fuseau horaire à utiliser avec une expression TIMESTAMP ou TIMESTAMPTZ.	AT TIME ZONE 'timezone'	TIMESTAMP ou TIMESTAMPZ
CONVERT_TIMEZONE Convertit un horodatage d'un fuseau horaire à un autre.	CONVERT_TIMEZONE (['timezone',] 'timezone', timestamp)	TIMESTAMP
CURRENT_DATE Renvoie une date selon le fuseau horaire de la séance en cours (UTC par défaut) pour le début de la transaction en cours.	CURRENT_DATE	DATE
DATE_CMP Compare les deux dates et renvoie 0 si les dates sont identiques, 1 si date1 est ultérieure, et -1 si date2 est ultérieure.	DATE_CMP (date1, date2)	INTEGER
DATE_CMP_TIMESTAMP Compare une date à une heure et renvoie 0 si les valeurs sont identiques, 1 si date est ultérieure et -1 si timestamp est ultérieur.	DATE_CMP_TIMESTAMP (date, timestamp)	INTEGER

Fonction	Syntaxe	Renvoie
DATE_CMP_TIMESTAMPTZ Compare une date et un horodatage avec fuseau horaire et renvoie `0` si les valeurs sont identiques, 1 si date est ultérieure et `-1` si timestamptz est ultérieur.	DATE_CMP_TIMESTAMPTZ (date, timestamptz)	`INTEGER`
DATE_PART_YEAR Extrait l'année d'une date.	DATE_PART_YEAR (date)	`INTEGER`
DATEADD Incrémente une date ou une heure par un intervalle spécifié.	DATEADD (datepart, interval, {date|time|horaire|timestamp})	`TIMESTAMP` ou `TIME` ou `TIMETZ`
DATEDIFF Renvoie la différence entre deux dates ou heures pour une partie de la date donnée, comme un jour ou un mois.	DATEDIFF (datepart, {date|time|timetz|timestamp}, {date|time|timetz|timestamp})	`BIGINT`
DATE_PART Extrait une valeur de la partie date d'une date ou d'une heure.	DATE_PART (datepart, {date|timestamp})	`DOUBLE`
DATE_TRUNC Tronque un horodatage selon une partie de date.	DATE_TRUNC ('datepart', timestamp)	`TIMESTAMP`
EXTRACT Extrait une partie de date ou d'heure d'un timestamp, d'un timestamptz, d'un time ou d'un timetz.	EXTRACT (datepart FROM source)	`INTEGER or DOUBLE`

Fonction	Syntaxe	Renvoie
GETDATE Renvoie la date et l'heure actuelles selon le fuseau horaire en cours (UTC par défaut). Les parenthèses sont obligatoires.	GETDATE()	TIMESTAMP
INTERVAL_CMP Compare deux intervalles et renvoie 0 si les intervalles sont identiques, 1 si interval1 est supérieur, et -1 si interval2 est supérieur.	INTERVAL_CMP (interval1, interval2)	INTEGER
LAST_DAY Renvoie la date du dernier jour du mois qui contient date.	LAST_DAY(date)	DATE
MONTHS_BETWEEN Renvoie le nombre de mois entre deux dates.	MONTHS_BETWEEN (date, date)	FLOAT8
NEXT_DAY Renvoie la date de la première instance de day ultérieure à date.	NEXT_DAY (date, day)	DATE
SYSDATE Renvoie la date et l'heure en UTC pour le début de la transaction en cours.	SYSDATE	TIMESTAMP
TIMEOFDAY Renvoie le jour de la semaine, et la date et l'heure actuelles selon le fuseau horaire en cours (UTC par défaut) sous la forme d'une valeur de chaîne.	TIMEOFDAY()	VARCHAR

Fonction	Syntaxe	Renvoie
TIMESTAMP_CMP Compare deux horodatages et renvoie 0 si les horodatages sont identiques, 1 si timestamp1 est supérieur et -1 si timestamp2 est supérieur.	TIMESTAMP_CMP (timestamp1, timestamp2)	INTEGER
TIMESTAMP_CMP_DATE Compare un horodatage à une date et renvoie 0 si les valeurs sont identiques, 1 si timestamp est ultérieur et -1 si date est ultérieure.	TIMESTAMP_CMP_DATE (timestamp, date)	INTEGER
TIMESTAMP_CMP_TIMESTAMPTZ Compare un horodatage avec un horodatage avec fuseau horaire et renvoie 0 si les valeurs sont égales, 1 si timestamp est supérieur et -1 si timestamptz est supérieur.	TIMESTAMP_CMP_TIMESTAMPTZ (timestamp, timestamptz)	INTEGER
TIMESTAMPTZ_CMP Compare deux valeurs d'horodatage avec fuseau horaire et renvoie 0 si les valeurs sont égales, 1 si timestamptz1 est supérieur et -1 si timestamptz2 est supérieur.	TIMESTAMPTZ_CMP (timestamptz1, timestamptz2)	INTEGER
TIMESTAMPTZ_CMP_DATE Compare la valeur d'un horodatage avec fuseau horaire et une date, puis renvoie 0 si les valeurs sont égales, 1 si timestamptz est supérieur et -1 si date est supérieure.	TIMESTAMPTZ_CMP_DATE (timestamptz, date)	INTEGER

Fonction	Syntaxe	Renvoie
TIMESTAMPTZ_CMP_TIMESTAMP Compare un horodatage avec fuseau horaire avec un horodatage et renvoie 0 si les valeurs sont égales, 1 si timestamptz est supérieur et -1 si timestamp est supérieur.	TIMESTAMPTZ_CMP_TIMESTAMP (timestamptz, timestamp)	INTEGER
TIMEZONE Renvoie un horodatage pour la valeur du fuseau horaire et de l'horodatage spécifiée.	TIMEZONE ('timezone' { timestamp | timestamptz)	TIMESTAMP ou TIMESTAMPTZ
TO_TIMESTAMP Renvoie un horodatage avec fuseau horaire pour le format de l'horodatage et du fuseau horaire spécifié.	TO_TIMESTAMP ('timestamp', 'format')	TIMESTAMPTZ
TRUNC Tronque un horodatage et renvoie une date.	TRUNC(timestamp)	DATE

> **Note**
>
> Les secondes supplémentaires ne sont pas prises en compte dans le calcul de durée écoulée.

Fonctions date et heure dans les transactions

Lorsque vous exécutez les fonctions suivantes au sein d'un bloc de transaction (BEGIN … END), la fonction renvoie la date ou l'heure de début de la transaction en cours, pas le début de l'instruction en cours.

- SYSDATE
- TIMESTAMP

- CURRENT_DATE

Les fonctions suivantes renvoient toujours la date ou l'heure de début de l'instruction en cours, même si elles se trouvaient sur un bloc de transaction.

- GETDATE
- TIMEOFDAY

Fonctions de nœud principal uniquement obsolètes

Les fonctions de date suivantes sont obsolètes, car elles s'exécutent uniquement sur le nœud principal. Pour plus d'informations, consultez Fonctions exécutées uniquement sur le nœud principal.

- AGE. Utilisez Fonction DATEDIFF à la place.
- CURRENT_TIME. Utilisez Fonction GETDATE ou SYSDATE à la place.
- CURRENT_TIMESTAMP. Utilisez Fonction GETDATE ou SYSDATE à la place.
- LOCALTIME. Utilisez Fonction GETDATE ou SYSDATE à la place.
- LOCALTIMESTAMP. Utilisez Fonction GETDATE ou SYSDATE à la place.
- ISFINITE
- NOW. Utilisez Fonction GETDATE ou SYSDATE à la place. Si vous utilisez la fonction NOW dans une vue matérialisée, elle utilise l'horodatage de la création de la vue matérialisée, au lieu de l'horodatage actuel.

+ Opérateur (concaténation)

Concatène une valeur DATE à une valeur TIME ou TIMETZ de chaque côté du symbole + et renvoie une valeur TIMESTAMP ou TIMESTAMPTZ.

Syntaxe

```
date + {time | timetz}
```

L'ordre des arguments peut être inversé. Par exemple, time + date.

Arguments

date

Colonne de type de données DATE ou expression implicitement évaluée à un type DATE.

time

Colonne de type de données TIME ou expression implicitement évaluée à un type TIME.

timetz

Colonne de type de données TIMETZ ou expression implicitement évaluée à un type TIMETZ.

Type de retour

TIMESTAMP si l'entrée est date + time.

TIMESTAMP si l'entrée est date + timetz.

Exemples

Exemple de configuration

Pour configurer les tables TIME_TEST et TIMETZ_TEST utilisées dans les exemples, utilisez la commande suivante.

```
create table time_test(time_val time);

insert into time_test values
('20:00:00'),
('00:00:00.5550'),
('00:58:00');

create table timetz_test(timetz_val timetz);

insert into timetz_test values
('04:00:00+00'),
('00:00:00.5550+00'),
('05:58:00+00');
```

Exemples avec une colonne time

L'exemple de table TIME_TEST suivant comporte une colonne TIME_VAL (type TIME) avec trois valeurs insérées.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
00:58:00
```

L'exemple suivant concatène une valeur de date littérale et une colonne TIME_VAL.

```
select date '2000-01-02' + time_val as ts from time_test;

ts
---------------------
2000-01-02 20:00:00
2000-01-02 00:00:00.5550
2000-01-02 00:58:00
```

L'exemple suivant concatène une valeur de date littérale et une valeur de temps littérale.

```
select date '2000-01-01' + time '20:00:00' as ts;

         ts
---------------------
 2000-01-01 20:00:00
```

L'exemple suivant concatène un littéral time et un littéral date.

```
select time '20:00:00' + date '2000-01-01' as ts;

         ts
---------------------
 2000-01-01 20:00:00
```

Exemples avec une colonne TIMETZ

L'exemple de table TIMETZ_TEST suivant comporte une colonne TIMETZ_VAL (type TIMETZ) avec trois valeurs insérées.

```
select timetz_val from timetz_test;
```

```
timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

L'exemple suivant concatène une valeur de date littérale et une colonne TIMETZ_VAL.

```
select date '2000-01-01' + timetz_val as ts from timetz_test;
ts
---------------------
2000-01-01 04:00:00+00
2000-01-01 00:00:00.5550+00
2000-01-01 05:58:00+00
```

L'exemple suivant concatène une colonne TIMETZ_VAL et un littéral date.

```
select timetz_val + date '2000-01-01' as ts from timetz_test;
ts
---------------------
2000-01-01 04:00:00+00
2000-01-01 00:00:00.5550+00
2000-01-01 05:58:00+00
```

L'exemple suivant concatène une valeur DATE littérale et une valeur TIMETZ littérale. L'exemple renvoie une valeur TIMESTAMPTZ qui se trouve dans le fuseau horaire UTC par défaut. Le fuseau horaire UTC ayant 8 heures d'avance sur PST, le résultat affiche 8 heures d'avance par rapport à l'heure d'entrée.

```
select date '2000-01-01' + timetz '20:00:00 PST' as ts;

          ts
------------------------
 2000-01-02 04:00:00+00
```

Fonction ADD_MONTHS

ADD_MONTHS ajoute le nombre de mois spécifié à une date, à une valeur d'horodatage ou à une expression. La fonction DATEADD fournit une fonctionnalité similaire.

Syntaxe

```
ADD_MONTHS( {date | timestamp}, integer)
```

Arguments

date | timestamp

Colonne de type de données DATE ou TIMESTAMP ou expression implicitement évaluée à un type DATE ou TIMESTAMP. Si la date est le dernier jour du mois, ou si le mois résultant est plus court, la fonction renvoie le dernier jour du mois dans le résultat. Pour les autres dates, le résultat contient le même nombre de jours que l'expression de date.

integer

Valeur du type de données INTEGER. Utilisez un nombre négatif pour soustraire des mois à partir de dates.

Type de retour

TIMESTAMP

Exemples

La requête suivante utilise la fonction ADD_MONTHS à l'intérieur d'une fonction TRUNC. La fonction TRUNC supprime l'heure du jour des résultats de ADD_MONTHS. La fonction ADD_MONTHS ajoute 12 mois à chaque valeur de la colonne CALDATE. Les valeurs de la colonne CALDATE sont des dates.

```
select distinct trunc(add_months(caldate, 12)) as calplus12,
trunc(caldate) as cal
from date
order by 1 asc;

 calplus12  |    cal
------------+------------
 2009-01-01 | 2008-01-01
 2009-01-02 | 2008-01-02
 2009-01-03 | 2008-01-03
...
(365 rows)
```

L'exemple suivant utilise la fonction ADD_MONTHS pour ajouter un mois à un timestamp (horodatage).

```
select add_months('2008-01-01 05:07:30', 1);

add_months
---------------------
2008-02-01 05:07:30
```

Les exemples suivants illustrent le comportement lorsque la fonction ADD_MONTHS opère sur des dates comportant des mois avec un nombre de jours différent. Cet exemple montre comment la fonction procède pour ajouter un mois au 31 mars et un mois au 30 avril. Sachant que le mois d'avril compte 30 jours, l'ajout d'un mois au 31 mars donne en résultat le 30 avril. En revanche, comme le mois de mai compte 31 jours, l'ajout d'un mois au 30 avril a pour résultat le 31 mai.

```
select add_months('2008-03-31',1);

add_months
---------------------
2008-04-30 00:00:00

select add_months('2008-04-30',1);

add_months
---------------------
2008-05-31 00:00:00
```

Fonction AT TIME ZONE

AT TIME ZONE spécifie le fuseau horaire à utiliser avec une expression TIMESTAMP ou TIMESTAMPTZ.

Syntaxe

```
AT TIME ZONE 'timezone'
```

Arguments

timezone

TIMEZONE pour la valeur renvoyée. Le fuseau horaire peut être spécifié comme nom de fuseau horaire (tel que **'Africa/Kampala'** ou **'Singapore'**) ou comme abréviation de fuseau horaire (telle que **'UTC'** ou **'PDT'**).

Pour afficher la liste des noms de fuseaux horaires pris en charge, exécutez la commande suivante.

```
select pg_timezone_names();
```

Pour afficher la liste des abréviations de fuseaux horaires prises en charge, exécutez la commande suivante.

```
select pg_timezone_abbrevs();
```

Pour plus d'informations et d'exemples, consultez Remarques sur l'utilisation de fuseaux horaires.

Type de retour

TIMESTAMPTZ lorsqu'il est utilisé avec une expression TIMESTAMP. TIMESTAMP lorsqu'il est utilisé avec une expression TIMESTAMPTZ.

Exemples

L'exemple suivant convertit une valeur d'horodatage sans fuseau horaire et l'interprète en tant qu'heure MST (UTC+7 dans POSIX). L'exemple renvoie une valeur dont le type de données est TIMESTAMPTZ pour le fuseau horaire UTC. Si vous configurez votre fuseau horaire par défaut sur un autre fuseau horaire qu'UTC, le résultat peut être différent.

```
SELECT TIMESTAMP '2001-02-16 20:38:40' AT TIME ZONE 'MST';

timezone
------------------------
2001-02-17 03:38:40+00
```

L'exemple suivant prend un horodatage d'entrée avec une valeur de fuseau horaire où le fuseau horaire spécifié est EST (UTC+5 dans POSIX) et le convertit en heure MST (UTC+7 dans POSIX). L'exemple renvoie une valeur dont le type de données est TIMESTAMP.

```
SELECT TIMESTAMPTZ '2001-02-16 20:38:40-05' AT TIME ZONE 'MST';

timezone
-----------------------
2001-02-16 18:38:40
```

Fonction CONVERT_TIMEZONE

CONVERT_TIMEZONE convertit un horodatage d'un fuseau horaire à un autre. La fonction s'ajuste automatiquement à l'heure d'été.

Syntaxe

```
CONVERT_TIMEZONE( ['source_timezone',] 'target_timezone', 'timestamp')
```

Arguments

source_timezone

(Facultatif) Fuseau horaire de l'horodatage actuel. La valeur par défaut est UTC. Pour plus d'informations, consultez Remarques sur l'utilisation de fuseaux horaires.

target_timezone

Fuseau horaire du nouvel horodatage. Pour plus d'informations, consultez Remarques sur l'utilisation de fuseaux horaires.

timestamp

Colonne timestamp ou expression qui convertit implicitement en un horodatage.

Type de retour

TIMESTAMP

Remarques sur l’utilisation de fuseaux horaires

source_timezone ou target_timezone peuvent être spécifiés sous forme de nom de fuseau horaire (tel que « Africa/Kampala » ou « Singapour ») ou d'abréviation de fuseau horaire (telle que « UTC » ou « PDT »). Il n'est pas nécessaire de convertir les noms de fuseaux horaires en noms ou les abréviations en abréviations. Par exemple, vous pouvez choisir un horodatage à partir du nom du fuseau horaire source « Singapour » et le convertir en horodatage dans l'abréviation du fuseau horaire « PDT ».

> **Note**
>
> Les résultats de l'utilisation d'un nom de fuseau horaire ou d'une abréviation de fuseau horaire peuvent être différents en fonction de l'heure saisonnière locale, telle que l'heure d'été.

Utilisation d’un nom de fuseau horaire

Pour afficher la liste complète et actuelle des noms de fuseaux horaires, exécutez la commande suivante.

```
select pg_timezone_names();
```

Chaque ligne contient une chaîne séparée par des virgules avec le nom du fuseau horaire, l'abréviation, le décalage UTC et un indicateur si le fuseau horaire respecte l'heure d'été (`touf`). Par exemple, l'extrait suivant montre deux lignes résultantes. La première ligne indique le fuseau horaire`Antarctica/South Pole`, abréviation`NZDT`, `13:00:00` décalé par rapport `f` à l'UTC, et indique que l'heure d'été n'est pas respectée. La deuxième ligne correspond au fuseau horaire`Europe/Paris`, abréviation`CET`, avec `01:00:00` décalage par rapport à l'heure UTC, et `f` pour indiquer qu'il respecte l'heure d'été.

```
pg_timezone_names
------------------
(Antarctica/South_Pole,NZDT,13:00:00,t)
(Europe/Paris,CET,01:00:00,f)
```

Exécutez l'instruction SQL pour obtenir la liste complète et trouver un nom de fuseau horaire. Environ 600 lignes sont renvoyées. Bien que certains des noms de fuseaux horaires renvoyés sont des sigles en majuscules ou des acronymes (par exemple : GB, PRC, ROK, etc.), la fonction

CONVERT_TIMEZONE les traite comme des noms de fuseaux horaires, pas comme des abréviations de fuseaux horaires.

Si vous spécifiez un fuseau horaire à l'aide d'un nom de fuseau horaire, CONVERT_TIMEZONE s'ajuste automatiquement à l'heure d'été (DST) ou à tout autre protocole saisonnier local, tel que l'heure d'été, l'heure standard ou l'heure d'hiver, qui est en vigueur pour ce fuseau horaire à la date et à l'heure spécifiées par « horodatage ». Par exemple, « Europe/Londres » représente l’heure UTC en hiver et une heure est ajoutée en été.

Utilisation d’une abréviation du fuseau horaire

Pour afficher la liste actuelle et complète des abréviations des fuseaux horaires, exécutez la commande suivante.

```
select pg_timezone_abbrevs();
```

Les résultats contiennent une chaîne séparée par des virgules avec l'abréviation du fuseau horaire, le décalage UTC et un indicateur si le fuseau horaire respecte l'heure d'été (`touf`). Par exemple, l'extrait suivant montre deux lignes résultantes. La première ligne contient l'abréviation de Pacific Daylight TimePDT, avec un `-07:00:00` décalage par rapport à l'UTC, et `t` pour indiquer qu'elle respecte l'heure d'été. La deuxième ligne contient l'abréviation de l'heure normale du PacifiquePST, `-08:00:00` décalée par rapport à l'heure UTC, `f` pour indiquer que l'heure d'été n'est pas respectée.

```
pg_timezone_abbrevs
--------------------
(PDT,-07:00:00,t)
(PST,-08:00:00,f)
```

Exécutez l'instruction SQL pour obtenir la liste complète et trouvez une abréviation en fonction de son indicateur de décalage et d'heure d'été. Environ 200 lignes sont renvoyées.

Les abréviations de fuseaux horaires représentent un décalage fixe de l’UTC. Si vous spécifiez un fuseau horaire à l'aide d'une abréviation de fuseau horaire, CONVERT_TIMEZONE utilise le décalage fixe par rapport à l'UTC et ne s'ajuste à aucun protocole saisonnier local.

Utilisation du format POSIX-Style

Une spécification de fuseau horaire de style POSIX se présente sous la forme STDoffsetSTDoffsetDST, où STD est une abréviation de fuseau horaire, offset est le décalage

numérique en heures à l'ouest par rapport à l'UTC et DST est une abréviation de zone d'été facultative. L'heure d'été est supposée avoir une heure d'avance sur le décalage indiqué.

Les formats de fuseaux horaires de style POSIX utilisent des décalages positifs à l'ouest de Greenwich, contrairement à la convention ISO-8601, qui utilise des décalages positifs à l'est de Greenwich.

Voici des exemples de fuseaux horaires de style POSIX :

- PST8
- PST8PDT
- EST5
- EST5EDT

> **Note**
>
> Amazon Redshift ne valide pas les spécifications de fuseaux horaires de style POSIX, il est donc possible définir le fuseau horaire sur une valeur non valide. Par exemple, la commande suivante ne renvoie pas d'erreur, même si elle définit le fuseau horaire sur une valeur non valide.
>
> ```
> set timezone to 'xxx36';
> ```

Exemples

Bon nombre d'exemples utilisent l'exemple d'ensemble de données TICKIT. Pour plus d'informations, consultez Exemple de base de données.

L'exemple suivant convertit la valeur d'horodatage du fuseau horaire UTC par défaut en HNP.

```
select convert_timezone('PST', '2008-08-21 07:23:54');

 convert_timezone
------------------------
2008-08-20 23:23:54
```

L'exemple suivant convertit la valeur d'horodatage dans la colonne LISTTIME du fuseau horaire UTC par défaut en HNP. Même si l'horodatage est à l'heure d'été, il est converti en heure normale, car le fuseau horaire cible est spécifié comme abréviation (PST).

```
select listtime, convert_timezone('PST', listtime) from listing
where listid = 16;

     listtime       |   convert_timezone
--------------------+-------------------
2008-08-24 09:36:12     2008-08-24 01:36:12
```

L'exemple suivant convertit une colonne LISTTIME d'horodatage du fuseau horaire UTC par défaut en fuseau horaire des États-Unis/Pacifique. Le fuseau horaire cible utilise un nom de fuseau horaire, et l'horodatage se situe pendant la période l'heure d'été, donc la fonction renvoie l'heure.

```
select listtime, convert_timezone('US/Pacific', listtime) from listing
where listid = 16;

     listtime       |   convert_timezone
--------------------+---------------------
2008-08-24 09:36:12 | 2008-08-24 02:36:12
```

L'exemple suivant convertit une chaîne d'horodatage de l'EST à PST :

```
select convert_timezone('EST', 'PST', '20080305 12:25:29');

 convert_timezone
-------------------
2008-03-05 09:25:29
```

L'exemple suivant convertit un horodatage à l'heure normale de l'est des États-Unis, car le fuseau horaire cible utilise un nom de fuseau horaire (Amérique/New_York) et que l'horodatage est à l'heure normale.

```
select convert_timezone('America/New_York', '2013-02-01 08:00:00');

 convert_timezone
---------------------
2013-02-01 03:00:00
(1 row)
```

L'exemple suivant convertit un horodatage à l'heure d'été de l'est des États-Unis, car le fuseau horaire cible utilise un nom de fuseau horaire (Amérique/New_York) et que l'horodatage est à l'heure d'été.

```
select convert_timezone('America/New_York', '2013-06-01 08:00:00');

 convert_timezone
---------------------
2013-06-01 04:00:00
(1 row)
```

L'exemple suivant illustre l'utilisation des décalages.

```
SELECT CONVERT_TIMEZONE('GMT','NEWZONE +2','2014-05-17 12:00:00') as newzone_plus_2,
CONVERT_TIMEZONE('GMT','NEWZONE-2:15','2014-05-17 12:00:00') as newzone_minus_2_15,
CONVERT_TIMEZONE('GMT','America/Los_Angeles+2','2014-05-17 12:00:00') as la_plus_2,
CONVERT_TIMEZONE('GMT','GMT+2','2014-05-17 12:00:00') as gmt_plus_2;

   newzone_plus_2    | newzone_minus_2_15  |      la_plus_2      |     gmt_plus_2
---------------------+---------------------+---------------------+---------------------
2014-05-17 10:00:00 | 2014-05-17 14:15:00 | 2014-05-17 10:00:00 | 2014-05-17 10:00:00
(1 row)
```

Fonction CURRENT_DATE

CURRENT_DATE renvoie une date dans le fuseau horaire de la session en cours (UTC par défaut) au format par défaut :. YYYY-MM-DD

> **Note**
>
> CURRENT_DATE renvoie la date de début de la transaction en cours, pas le début de l'instruction en cours. Imaginez un scénario où vous démarrez une transaction contenant plusieurs instructions le 10/01/08 à 23h59, et où l'instruction contenant CURRENT_DATE s'exécute le 10/02/08 à 00h00. CURRENT_DATE renvoie `10/01/08`, et non `10/02/08`.

Syntaxe

```
CURRENT_DATE
```

Type de retour

DATE

Exemples

L'exemple suivant renvoie la date actuelle (dans Région AWS laquelle la fonction s'exécute).

```
select current_date;

   date
------------
2008-10-01
```

Dans l'exemple suivant, un tableau est créé, une ligne est insérée, et la valeur par défaut de la colonne `todays_date` est CURRENT_DATE. Enfin, toutes les lignes du tableau sont sélectionnées.

```
CREATE TABLE insert_dates(
    label varchar(128) NOT NULL,
    todays_date DATE DEFAULT CURRENT_DATE);

INSERT INTO insert_dates(label)
VALUES('Date row inserted');

SELECT * FROM insert_dates;


 label            | todays_date
------------------+-------------
Date row inserted | 2023-05-10
```

Fonction DATE_CMP

DATE_CMP compare deux dates. La fonction renvoie `0` si les dates sont identiques, 1 si date1 est ultérieure, et `-1` si date2 est ultérieure.

Syntaxe

```
DATE_CMP(date1, date2)
```

Arguments

date1

Colonne de type de données DATE ou expression qui est évaluée sur un type DATE.

date2

Colonne de type de données DATE ou expression qui est évaluée sur un type DATE.

Type de retour

INTEGER

Exemples

La requête suivante compare les valeurs de DATE de la colonne CALDATE à la date du 4 janvier 2008 et indique si la valeur de CALDATE est antérieure (-1), égale (0) ou postérieure au 4 janvier 2008 :

```
select caldate, '2008-01-04',
date_cmp(caldate,'2008-01-04')
from date
order by dateid
limit 10;

 caldate   |  ?column?  | date_cmp
-----------+------------+----------
2008-01-01 | 2008-01-04 |       -1
2008-01-02 | 2008-01-04 |       -1
2008-01-03 | 2008-01-04 |       -1
2008-01-04 | 2008-01-04 |        0
2008-01-05 | 2008-01-04 |        1
2008-01-06 | 2008-01-04 |        1
2008-01-07 | 2008-01-04 |        1
2008-01-08 | 2008-01-04 |        1
2008-01-09 | 2008-01-04 |        1
2008-01-10 | 2008-01-04 |        1
(10 rows)
```

Fonction DATE_CMP_TIMESTAMP

DATE_CMP_TIMESTAMP compare une date à un timestamp (horodatage) et renvoie `0` si les valeurs sont identiques, 1 si la valeur de date est supérieure du point de vue chronologique et `-1` si la valeur de timestamp est supérieure.

Syntaxe

```
DATE_CMP_TIMESTAMP(date, timestamp)
```

Arguments

date

Colonne de type de données `DATE` ou expression qui est évaluée sur un type `DATE`.

timestamp

Colonne de type de données `TIMESTAMP` ou expression qui est évaluée sur un type `TIMESTAMP`.

Type de retour

INTEGER

Exemples

L'exemple suivant compare la date `2008-06-18` à LISTTIME. Les valeurs de la colonne LISTTIME sont des horodatages. Les listes faites avant cette date renvoient 1 ; les listes faites après cette date renvoient `-1`.

```
select listid, '2008-06-18', listtime,
date_cmp_timestamp('2008-06-18', listtime)
from listing
order by 1, 2, 3, 4
limit 10;

 listid |  ?column?  |      listtime       | date_cmp_timestamp
--------+------------+---------------------+--------------------
      1 | 2008-06-18 | 2008-01-24 06:43:29 |                  1
      2 | 2008-06-18 | 2008-03-05 12:25:29 |                  1
      3 | 2008-06-18 | 2008-11-01 07:35:33 |                 -1
      4 | 2008-06-18 | 2008-05-24 01:18:37 |                  1
      5 | 2008-06-18 | 2008-05-17 02:29:11 |                  1
```

```
      6 | 2008-06-18 | 2008-08-15 02:08:13 |                -1
      7 | 2008-06-18 | 2008-11-15 09:38:15 |                -1
      8 | 2008-06-18 | 2008-11-09 05:07:30 |                -1
      9 | 2008-06-18 | 2008-09-09 08:03:36 |                -1
     10 | 2008-06-18 | 2008-06-17 09:44:54 |                 1
(10 rows)
```

Fonction DATE_CMP_TIMESTAMPTZ

DATE_CMP_TIMESTAMPTZ compare une date à un timestamp (horodatage) avec fuseau horaire et renvoie `0` si les valeurs sont identiques, 1 si la valeur de date est supérieure du point de vue chronologique et `-1` si la valeur de timestamptz est supérieure.

Syntaxe

```
DATE_CMP_TIMESTAMPTZ(date, timestamptz)
```

Arguments

date

Colonne de type de données `DATE` ou expression implicitement évaluée à un type `DATE`.

timestamptz

Colonne de type de données `TIMESTAMPTZ` ou expression implicitement évaluée à un type `TIMESTAMPTZ`.

Type de retour

INTEGER

Exemples

L'exemple suivant compare la date `2008-06-18` à LISTTIME. Les listes faites avant cette date renvoient 1 ; les listes faites après cette date renvoient `-1`.

```
select listid, '2008-06-18', CAST(listtime AS timestamptz),
date_cmp_timestamptz('2008-06-18', CAST(listtime AS timestamptz))
from listing
order by 1, 2, 3, 4
limit 10;
```

```
 listid |  ?column?  |      timestamptz       | date_cmp_timestamptz
--------+------------+------------------------+----------------------
      1 | 2008-06-18 | 2008-01-24 06:43:29+00 |                    1
      2 | 2008-06-18 | 2008-03-05 12:25:29+00 |                    1
      3 | 2008-06-18 | 2008-11-01 07:35:33+00 |                   -1
      4 | 2008-06-18 | 2008-05-24 01:18:37+00 |                    1
      5 | 2008-06-18 | 2008-05-17 02:29:11+00 |                    1
      6 | 2008-06-18 | 2008-08-15 02:08:13+00 |                   -1
      7 | 2008-06-18 | 2008-11-15 09:38:15+00 |                   -1
      8 | 2008-06-18 | 2008-11-09 05:07:30+00 |                   -1
      9 | 2008-06-18 | 2008-09-09 08:03:36+00 |                   -1
     10 | 2008-06-18 | 2008-06-17 09:44:54+00 |                    1
(10 rows)
```

Fonction DATEADD

Augmente une valeur DATE, TIME, TIMETZ ou TIMESTAMP d'un intervalle spécifié.

Syntaxe

```
DATEADD( datepart, interval, {date|time|timetz|timestamp} )
```

Arguments

datepart

Partie de la date (par exemple, année, mois, jour ou heure) sur laquelle la fonction opère. Pour plus d'informations, consultez Parties de date pour les fonctions de date ou d'horodatage.

interval

Nombre entier qui a spécifié l'intervalle (nombre de jours, par exemple) à ajouter à l'expression cible. Un nombre entier négatif soustrait l'intervalle.

date|time|timetz|timestamp

Colonne DATE, TIME, TIMETZ ou TIMESTAMP, ou expression qui convertit implicitement en un horodatage ou valeur DATE, TIME, TIMETZ ou TIMESTAMP. L'expression DATE, TIME, TIMETZ ou TIMESTAMP doit contenir la partie de date spécifiée.

Type de retour

TIMESTAMP ou TIME ou TIMETZ selon le type de données d'entrée.

Exemples avec une colonne DATE

Dans l'exemple suivant, 30 jours sont ajoutés à chaque date en novembre qui existe dans la table DATE.

```
select dateadd(day,30,caldate) as novplus30
from date
where month='NOV'
order by dateid;

novplus30
---------------------
2008-12-01 00:00:00
2008-12-02 00:00:00
2008-12-03 00:00:00
...
(30 rows)
```

L'exemple suivant ajoute 18 mois à une valeur de date littérale.

```
select dateadd(month,18,'2008-02-28');

date_add
---------------------
2009-08-28 00:00:00
(1 row)
```

Le nom de colonne par défaut pour une fonction DATEADD est DATE_ADD. L'horodatage par défaut pour une valeur de date est `00:00:00`.

L'exemple suivant ajoute 30 minutes à une valeur de date qui ne spécifie pas d'horodatage.

```
select dateadd(m,30,'2008-02-28');

date_add
---------------------
2008-02-28 00:30:00
(1 row)
```

Vous pouvez nommer les parties de date intégralement ou les abréger. Dans ce cas, m représente les minutes, et non les mois.

Exemples avec une colonne TIME

L'exemple de table TIME_TEST suivant comporte une colonne TIME_VAL (type TIME) avec trois valeurs insérées.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
00:58:00
```

L'exemple suivant ajoute 5 minutes à chaque TIME_VAL de la table TIME_TEST.

```
select dateadd(minute,5,time_val) as minplus5 from time_test;

minplus5
---------------
20:05:00
00:05:00.5550
01:03:00
```

L'exemple suivant ajoute 8 heures à une valeur de temps littérale.

```
select dateadd(hour, 8, time '13:24:55');

date_add
---------------
21:24:55
```

L'exemple suivant montre quand une heure est supérieure à 24:00:00 ou inférieure à 00:00:00.

```
select dateadd(hour, 12, time '13:24:55');

date_add
---------------
01:24:55
```

Exemples avec une colonne TIMETZ

Les valeurs de sortie de ces exemples utilisent le fuseau horaire par défaut UTC.

L'exemple de table TIMETZ_TEST suivant comporte une colonne TIMETZ_VAL (type TIMETZ) avec trois valeurs insérées.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

L’exemple suivant ajoute 5 minutes à chaque TIMETZ_VAL de la table TIMETZ_TEST.

```
select dateadd(minute,5,timetz_val) as minplus5_tz from timetz_test;

minplus5_tz
---------------
04:05:00+00
00:05:00.5550+00
06:03:00+00
```

L’exemple suivant ajoute 2 heures à une valeur timetz littérale.

```
select dateadd(hour, 2, timetz '13:24:55 PST');

date_add
---------------
23:24:55+00
```

Exemples avec une colonne TIMESTAMP

Les valeurs de sortie de ces exemples utilisent le fuseau horaire par défaut UTC.

L’exemple de table TIMESTAMP_TEST suivant comporte une colonne TIMESTAMP_VAL (type TIMESTAMP) avec trois valeurs insérées.

```
SELECT timestamp_val FROM timestamp_test;

timestamp_val
------------------
1988-05-15 10:23:31
2021-03-18 17:20:41
```

```
2023-06-02 18:11:12
```

L'exemple suivant ajoute 20 ans uniquement aux valeurs TIMESTAMP_VAL de TIMESTAMP_TEST antérieures à l'an 2000.

```
SELECT dateadd(year,20,timestamp_val)
FROM timestamp_test
WHERE timestamp_val < to_timestamp('2000-01-01 00:00:00', 'YYYY-MM-DD HH:MI:SS');

date_add
---------------
2008-05-15 10:23:31
```

L'exemple suivant ajoute 5 secondes à une valeur d'horodatage littérale écrite sans indicateur de secondes.

```
SELECT dateadd(second, 5, timestamp '2001-06-06');

date_add
---------------
2001-06-06 00:00:05
```

Notes d'utilisation

Les fonctions DATEADD(month, ...) et ADD_MONTHS gèrent les dates tombant différemment en fin de mois :

- ADD_MONTHS : Si la date que vous ajoutez est le dernier jour du mois, le résultat est toujours le dernier jour du mois du résultat, quelle que soit la longueur du mois. Par exemple, 30 avril + 1 mois est le 31 mai :

  ```
  select add_months('2008-04-30',1);

  add_months
  ---------------------
  2008-05-31 00:00:00
  (1 row)
  ```

- DATEADD : S'il y a moins de jours dans la date que vous ajoutez que dans le mois du résultat, le résultat sera le jour correspondant du mois du résultat, pas le dernier jour du mois. Par exemple, 30 avril + 1 mois est le 30 mai :

```
select dateadd(month,1,'2008-04-30');

date_add
---------------------
2008-05-30 00:00:00
(1 row)
```

La fonction DATEADD gère la date de l'année bissextile du 29/02 différemment selon que vous utilisez dateadd(month, 12,...) ou dateadd(year, 1,...).

```
select dateadd(month,12,'2016-02-29');

date_add
---------------------
2017-02-28 00:00:00

select dateadd(year, 1, '2016-02-29');

date_add
---------------------
2017-03-01 00:00:00
```

Fonction DATEDIFF

DATEDIFF renvoie la différence entre les parties de date de deux expressions de date ou d'heure.

Syntaxe

```
DATEDIFF( datepart, {date|time|timetz|timestamp}, {date|time|timetz|timestamp} )
```

Arguments

datepart

Partie spécifique de la valeur date ou time (année, mois, ou jour, heure, minute, seconde, milliseconde ou microseconde) sur laquelle la fonction opère. Pour plus d'informations, consultez Parties de date pour les fonctions de date ou d'horodatage.

Plus précisément, DATEDIFF détermine le nombre de limites de partie de date qui sont traversées entre deux expressions. Par exemple, supposons que vous calculez la différence

en années entre deux dates, `12-31-2008` et `01-01-2009`. Dans ce cas, la fonction renvoie 1 an malgré le fait que ces dates n'ont qu'un jour d'écart. Si vous trouvez la différence en heures entre les deux horodatages, `01-01-2009 8:30:00` et `01-01-2009 10:00:00`, le résultat est 2 heures. Si vous trouvez la différence en heures entre les deux horodatages, `8:30:00` et `10:00:00`, le résultat est 2 heures.

date|time|timetz|timestamp

Une colonne ou des expressions DATE, TIME, TIMETZ ou TIMESTAMP qui se convertissent implicitement en DATE, TIME, TIMETZ ou TIMESTAMP. Les expressions régulières doivent contenir la date ou partie de date spécifiée. Si la seconde date ou heure est ultérieure à la première date ou heure, le résultat est positif. Si la seconde date ou heure est antérieure à la première date ou heure, le résultat est négatif.

Type de retour

BIGINT

Exemples avec une colonne DATE

L'exemple suivant met en évidence la différence, en nombre de semaines, entre deux valeurs de date littérales.

```
select datediff(week,'2009-01-01','2009-12-31') as numweeks;

numweeks
----------
52
(1 row)
```

L'exemple suivant permet de trouver la différence, en heures, entre deux valeurs littérales de date. Si vous n'indiquez pas la valeur temporelle d'une date, celle-ci est fixée par défaut à 00:00:00.

```
select datediff(hour, '2023-01-01', '2023-01-03 05:04:03');

date_diff
----------
53
(1 row)
```

L'exemple suivant permet de trouver la différence, en jours, entre deux valeurs TIMESTAMETZ littérales.

```
Select datediff(days, 'Jun 1,2008  09:59:59 EST', 'Jul 4,2008  09:59:59 EST')

date_diff
----------
33
```

L'exemple suivant permet de trouver la différence, en jours, entre deux dates figurant sur la même ligne d'une table.

```
select * from date_table;

start_date |   end_date
-----------+-----------
2009-01-01 | 2009-03-23
2023-01-04 | 2024-05-04
(2 rows)

select datediff(day, start_date, end_date) as duration from date_table;

duration
---------
      81
     486
(2 rows)
```

L'exemple suivant met en évidence la différence, dans le nombre de trimestres, entre une valeur littérale dans le passé et la date du jour. Cet exemple suppose que la date du jour est le 5 juin 2008. Vous pouvez nommer les parties de date intégralement ou les abréger. Le nom de colonne par défaut pour la fonction DATEDIFF est DATE_DIFF.

```
select datediff(qtr, '1998-07-01', current_date);

date_diff
-----------
40
(1 row)
```

L’exemple suivant joint les tables SALES et LISTING pour calculer combien de jours après leur mise en vente des billets ont été vendus pour les listes 1000 à 1005. L’attente la plus longue pour les ventes de ces listes a été 15 jours, et la plus courte a été de moins d’une journée (0 jour).

```
select priceperticket,
datediff(day, listtime, saletime) as wait
from sales, listing where sales.listid = listing.listid
and sales.listid between 1000 and 1005
order by wait desc, priceperticket desc;

priceperticket | wait
---------------+------
 96.00         |   15
 123.00        |   11
 131.00        |    9
 123.00        |    6
 129.00        |    4
 96.00         |    4
 96.00         |    0
(7 rows)
```

Cet exemple calcule la moyenne du nombre d’heures que les vendeurs ont attendu pour toutes les ventes de billets.

```
select avg(datediff(hours, listtime, saletime)) as avgwait
from sales, listing
where sales.listid = listing.listid;

avgwait
---------
465
(1 row)
```

Exemples avec une colonne TIME

L'exemple de table TIME_TEST suivant comporte une colonne TIME_VAL (type TIME) avec trois valeurs insérées.

```
select time_val from time_test;

time_val
--------------------
```

```
20:00:00
00:00:00.5550
00:58:00
```

L'exemple suivant montre comment trouver la différence en nombre d'heures entre la colonne TIME_VAL et une valeur de temps littérale.

```
select datediff(hour, time_val, time '15:24:45') from time_test;

 date_diff
-----------
        -5
        15
        15
```

L'exemple suivant montre comment trouver la différence en nombre de minutes entre deux valeurs de temps littérales.

```
select datediff(minute, time '20:00:00', time '21:00:00') as nummins;

nummins
----------
60
```

Exemples avec une colonne TIMETZ

L'exemple de table TIMETZ_TEST suivant comporte une colonne TIMETZ_VAL (type TIMETZ) avec trois valeurs insérées.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

L'exemple suivant montre comment trouver la différence en nombre d'heures entre une valeur TIMETZ littérale et une valeur timez_val.

```
select datediff(hours, timetz '20:00:00 PST', timetz_val) as numhours from timetz_test;
```

```
numhours
----------
0
-4
1
```

L'exemple suivant montre comment trouver la différence en nombre d'heures entre deux valeurs TIMETZ littérales.

```
select datediff(hours, timetz '20:00:00 PST', timetz '00:58:00 EST') as numhours;

numhours
----------
1
```

Fonction DATE_PART

DATE_PART extrait des valeurs date part d'une expression. DATE_PART est un synonyme de la fonction PGDATE_PART.

Syntaxe

```
DATE_PART(datepart, {date|timestamp})
```

Arguments

datepart

Un identifiant littéral ou une chaîne de caractères de la partie spécifique de la valeur de la date (par exemple, l'année, le mois ou le jour) sur laquelle la fonction opère. Pour de plus amples informations, veuillez consulter Parties de date pour les fonctions de date ou d'horodatage.

{date|timestamp}

Une colonne de date, une colonne d'horodatage ou une expression qui se convertit implicitement en date ou en horodatage. La colonne ou l'expression en date ou timestamp doit contenir la partie date spécifiée dans datepart.

Type de retour

DOUBLE

Exemples

Le nom de colonne par défaut pour la fonction DATE_PART est `pgdate_part`.

Pour obtenir plus d'informations sur les données utilisées dans certains de ces exemples, consultez Exemple de base de données.

L'exemple suivant recherche le jour de la semaine à partir d'un littéral d'horodatage.

```
SELECT DATE_PART(minute, timestamp '20230104 04:05:06.789');

pgdate_part
-----------
          5
```

L'exemple suivant recherche le numéro de semaine à partir d'un littéral d'horodatage. Le calcul du numéro de semaine est conforme à la norme ISO 8601. Pour plus d'informations, consultez la page Wikipédia ISO 8601.

```
SELECT DATE_PART(week, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
         18
```

L'exemple suivant recherche le jour du mois à partir d'un littéral d'horodatage.

```
SELECT DATE_PART(day, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
          2
```

L'exemple suivant recherche le jour de la semaine à partir d'un littéral d'horodatage. Le jour de la semaine est un nombre entier compris entre 0 et 6, en commençant par dimanche.

```
SELECT DATE_PART(dayofweek, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
          1
```

L'exemple suivant recherche le siècle à partir d'un littéral d'horodatage. Le calcul du siècle suit la norme ISO 8601. Pour plus d'informations, consultez la page Wikipédia ISO 8601.

```
SELECT DATE_PART(century, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
         21
```

L'exemple suivant permet de trouver le millénaire à partir d'un littéral d'horodatage. Le calcul du millénaire suit la norme ISO 8601. Pour plus d'informations, consultez la page Wikipédia ISO 8601.

```
SELECT DATE_PART(millennium, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
         3
```

L'exemple suivant permet de trouver les microsecondes à partir d'un littéral d'horodatage. Le calcul des microsecondes est conforme à la norme ISO 8601. Pour plus d'informations, consultez la page Wikipédia ISO 8601.

```
SELECT DATE_PART(microsecond, timestamp '20220502 04:05:06.789');

pgdate_part
-----------
    789000
```

L'exemple suivant recherche le mois à partir d'un littéral de date.

```
SELECT DATE_PART(month, date '20220502');

pgdate_part
-----------
         5
```

L'exemple suivant applique la fonction DATE_PART à une colonne dans une table.

```
SELECT date_part(w, listtime) AS weeks, listtime
FROM listing
```

```
WHERE listid=10


weeks |      listtime
------+---------------------
 25   | 2008-06-17 09:44:54
(1 row)
```

Vous pouvez nommer les parties de date intégralement ou les abrégés ; dans ce cas, w est synonyme de semaines.

Le jour de la semaine renvoie un nombre entier compris entre 0 et 6, en commençant par dimanche. Utilisez DATE_PART avec dow (DAYOFWEEK) afin d'afficher les événements d'un samedi.

```
SELECT date_part(dow, starttime) AS dow, starttime
FROM event
WHERE date_part(dow, starttime)=6
ORDER BY 2,1;

 dow |      starttime
-----+---------------------
   6 | 2008-01-05 14:00:00
   6 | 2008-01-05 14:00:00
   6 | 2008-01-05 14:00:00
   6 | 2008-01-05 14:00:00
...
(1147 rows)
```

Fonction DATE_PART_YEAR

La fonction DATE_PART_YEAR extrait l'année d'une date.

Syntaxe

```
DATE_PART_YEAR(date)
```

Argument

date

Colonne de type de données DATE ou expression implicitement évaluée à un type DATE.

Type de retour

INTEGER

Exemples

L'exemple suivant recherche l'année à partir d'un littéral de date.

```
SELECT DATE_PART_YEAR(date '20220502 04:05:06.789');

date_part_year
---------------
2022
```

L'exemple suivant extrait l'année de la colonne CALDATE. Les valeurs de la colonne CALDATE sont des dates. Pour obtenir plus d'informations sur les données utilisées dans cet exemple, consultez Exemple de base de données.

```
select caldate, date_part_year(caldate)
from date
order by
dateid limit 10;

 caldate   | date_part_year
-----------+----------------
2008-01-01 |           2008
2008-01-02 |           2008
2008-01-03 |           2008
2008-01-04 |           2008
2008-01-05 |           2008
2008-01-06 |           2008
2008-01-07 |           2008
2008-01-08 |           2008
2008-01-09 |           2008
2008-01-10 |           2008
(10 rows)
```

Fonction DATE_TRUNC

La fonction DATE_TRUNC tronque une expression d'horodatage ou littérale en fonction de la partie de date que vous spécifiez, telle que l'heure, le jour ou le mois.

Syntaxe

```
DATE_TRUNC('datepart', timestamp)
```

Arguments

datepart

Partie de la date à laquelle tronquer la valeur d'horodatage. L'entrée timestamp est tronquée à la précision de l'entrée datepart. Par exemple, `month` tronque jusqu'au premier jour du mois. Les formats valides sont les suivants :

- microseconde, microsecondes
- milliseconde, millisecondes
- seconde, secondes
- minute, minutes
- heure, heures
- jour, jours
- semaine, semaines
- mois
- trimestre, trimestres
- année, années
- décennie, décennies
- siècle, siècles
- millénaire, millénaires

Pour plus d'informations sur les abréviations de certains formats, consultez Parties de date pour les fonctions de date ou d'horodatage.

timestamp

Colonne timestamp ou expression qui convertit implicitement en un horodatage.

Type de retour

TIMESTAMP

Exemples

Tronquer l'horodatage en entrée à la seconde.

```
SELECT DATE_TRUNC('second', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 04:05:06
```

Tronquer l'horodatage en entrée à la minute.

```
SELECT DATE_TRUNC('minute', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 04:05:00
```

Tronquer l'horodatage en entrée à l'heure.

```
SELECT DATE_TRUNC('hour', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 04:00:00
```

Tronquer l'horodatage en entrée au jour.

```
SELECT DATE_TRUNC('day', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-30 00:00:00
```

Tronquer l'horodatage en entrée au premier jour du mois.

```
SELECT DATE_TRUNC('month', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-01 00:00:00
```

Tronquer l'horodatage en entrée au premier jour d'un trimestre.

```
SELECT DATE_TRUNC('quarter', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-04-01 00:00:00
```

Tronquer l'horodatage en entrée au premier jour de l'année.

```
SELECT DATE_TRUNC('year', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2020-01-01 00:00:00
```

Tronquer l’horodatage en entrée au premier jour d’un siècle.

```
SELECT DATE_TRUNC('millennium', TIMESTAMP '20200430 04:05:06.789');
date_trunc
2001-01-01 00:00:00
```

Tronquez l’horodatage en entrée au lundi d’une semaine.

```
select date_trunc('week', TIMESTAMP '20220430 04:05:06.789');
date_trunc
2022-04-25 00:00:00
```

Dans l’exemple suivant, la fonction DATE_TRUNC utilise la partie de date ’week’ pour renvoyer la date du lundi de chaque semaine.

```
select date_trunc('week', saletime), sum(pricepaid) from sales where
saletime like '2008-09%' group by date_trunc('week', saletime) order by 1;

date_trunc  |    sum
------------+-------------
2008-09-01  | 2474899
2008-09-08  | 2412354
2008-09-15  | 2364707
2008-09-22  | 2359351
2008-09-29  |  705249
```

Fonction EXTRACT

La fonction EXTRACT renvoie une partie de date ou d'heure à partir d'une valeur TIMESTAMP, TIMESTAMPTZ, TIME, TIMETZ, INTERVAL YEAR TO MONTH ou INTERVAL DAY TO SECOND. Les exemples incluent le jour, le mois, l’année, l’heure, la minute, la seconde, la milliseconde ou la microseconde d’un horodatage.

Syntaxe

```
EXTRACT(datepart FROM source)
```

Arguments

datepart

Sous-champ d'une date ou d'une heure à extraire, tel que le jour, le mois, l'année, l'heure, la minute, la seconde, la milliseconde ou la microseconde. Pour les valeurs possibles, consultez Parties de date pour les fonctions de date ou d'horodatage.

source

Colonne ou expression dont le type de données est TIMESTAMP, TIMESTAMPTZ, TIME, TIMETZ, INTERVAL YEAR TO MONTH ou INTERVAL DAY TO SECOND.

Type de retour

INTEGER si la valeur source est le type de données TIMESTAMP, TIME, TIMETZ, INTERVAL YEAR TO MONTH ou INTERVAL DAY TO SECOND.

DOUBLE PRECISION si la valeur source est de type TIMESTAMPTZ.

Exemples avec TIMESTAMP

L'exemple suivant renvoie le nombre de semaines pour les ventes au cours desquelles le prix payé était de 10 000 $ ou plus. Cet exemple utilise les données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

```
select salesid, extract(week from saletime) as weeknum
from sales
where pricepaid > 9999
order by 2;

salesid | weeknum
--------+---------
 159073 |       6
 160318 |       8
 161723 |      26
```

L'exemple suivant renvoie la valeur de minute à partir d'une valeur d'horodatage littérale.

```
select extract(minute from timestamp '2009-09-09 12:08:43');

date_part
```

```
-----------
8
```

L'exemple suivant renvoie la valeur de la milliseconde à partir d'une valeur littérale d'horodatage.

```
select extract(ms from timestamp '2009-09-09 12:08:43.101');

date_part
-----------
101
```

Exemples avec TIMESTAMPTZ

L'exemple suivant renvoie la valeur de l'année à partir d'une valeur littérale de timestamptz.

```
select extract(year from timestamptz '1.12.1997 07:37:16.00 PST');

date_part
-----------
1997
```

Exemples avec TIME

L'exemple de table TIME_TEST suivant comporte une colonne TIME_VAL (type TIME) avec trois valeurs insérées.

```
select time_val from time_test;

time_val
---------------------
20:00:00
00:00:00.5550
00:58:00
```

L'exemple suivant extrait les minutes de chaque time_val.

```
select extract(minute from time_val) as minutes from time_test;

minutes
-----------
         0
```

```
         0
         58
```

L’exemple suivant extrait les heures de chaque time_val.

```
select extract(hour from time_val) as hours from time_test;

hours
-----------
         20
         0
         0
```

L’exemple suivant extrait des millisecondes d’une valeur littérale.

```
select extract(ms from time '18:25:33.123456');

 date_part
-----------
     123
```

Exemples avec TIMETZ

L'exemple de table TIMETZ_TEST suivant comporte une colonne TIMETZ_VAL (type TIMETZ) avec trois valeurs insérées.

```
select timetz_val from timetz_test;

timetz_val
------------------
04:00:00+00
00:00:00.5550+00
05:58:00+00
```

L'exemple suivant extrait les heures de chaque timez_val.

```
select extract(hour from timetz_val) as hours from time_test;

hours
-----------
         4
```

```
         0
         5
```

L'exemple suivant extrait des millisecondes d'une valeur littérale. Les valeurs littérales ne sont pas converties en UTC avant le traitement de l'extraction.

```
select extract(ms from timetz '18:25:33.123456 EST');

 date_part
-----------
     123
```

L'exemple suivant renvoie l'heure de décalage du fuseau horaire par rapport à UTC à partir d'une valeur littérale de timetz.

```
select extract(timezone_hour from timetz '1.12.1997 07:37:16.00 PDT');

date_part
-----------
-7
```

Exemples avec INTERVAL YEAR TO MONTH et INTERVAL DAY TO SECOND

L'exemple suivant extrait la partie du jour 1 de l'intervalle entre jours et secondes qui définit 36 heures, soit 1 jour et 12 heures.

```
select EXTRACT('days' from INTERVAL '36 hours' DAY TO SECOND)

 date_part
------------------
 1
```

L'exemple suivant extrait la partie mensuelle 3 de l'année au mois qui définit 15 mois, soit 1 an et 3 mois.

```
select EXTRACT('month' from INTERVAL '15 months' YEAR TO MONTH)

 date_part
------------------
```

```
 3
```

L'exemple suivant extrait la partie mensuelle 6 de 30 mois, soit 2 ans et 6 mois.

```
select EXTRACT('month' from INTERVAL '30' MONTH)

 date_part
------------------
 6
```

L'exemple suivant extrait la partie horaire 2 de 50 heures, soit 2 jours et 2 heures.

```
select EXTRACT('hours' from INTERVAL '50' HOUR)

 date_part
------------------
 2
```

L'exemple suivant extrait la partie minute 11 de 1 heure 11 minutes 11,123 secondes.

```
select EXTRACT('minute' from INTERVAL '70 minutes 70.123 seconds' MINUTE TO SECOND)

 date_part
------------------
 11
```

L'exemple suivant extrait la partie des secondes 1.11 de 1 jour 1 heure 1 minute 1,11 seconde.

```
select EXTRACT('seconds' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND)

 date_part
------------------
 1.11
```

L'exemple suivant extrait le nombre total d'heures d'un INTERVAL. Chaque partie est extraite et ajoutée à un total.

```
select EXTRACT('days' from INTERVAL '50' HOUR) * 24 + EXTRACT('hours' from INTERVAL
 '50' HOUR)
```

```
 ?column?
------------------
 50
```

L'exemple suivant extrait le nombre total de secondes d'un INTERVAL. Chaque partie est extraite et ajoutée à un total.

```
select EXTRACT('days' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND) * 86400 +
       EXTRACT('hours' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND) * 3600 +
       EXTRACT('minutes' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND) * 60 +
       EXTRACT('seconds' from INTERVAL '1 day 1:1:1.11' DAY TO SECOND)

 ?column?
------------------
 90061.11
```

Fonction GETDATE

GETDATE renvoie la date et l’heure actuelles selon le fuseau horaire en cours (UTC par défaut). Cette fonction renvoie la date ou l’heure de début de l’instruction actuelle, même lorsqu’elle se trouve dans un bloc de transaction.

Syntaxe

```
GETDATE()
```

Les parenthèses sont obligatoires.

Type de retour

TIMESTAMP

Exemples

L’exemple suivant utilise la fonction GETDATE pour renvoyer l’horodatage complet de la date du jour.

```
select getdate();

timestamp
---------------------
2008-12-04 16:10:43
```

L'exemple suivant utilise la fonction GETDATE à l'intérieur de la fonction TRUNC pour renvoyer la date du jour sans l'heure.

```
select trunc(getdate());

trunc
------------
2008-12-04
```

Fonction INTERVAL_CMP

INTERVAL_CMP compare deux intervalles et renvoie 1 si le premier intervalle est supérieur, `-1` si le deuxième intervalle est supérieur, et `0` si les intervalles sont égaux. Pour plus d'informations, consultez Exemples de littéraux d'intervalle sans syntaxe de qualificatif.

Syntaxe

```
INTERVAL_CMP(interval1, interval2)
```

Arguments

interval1

Valeur d'intervalle littérale.

interval2

Valeur d'intervalle littérale.

Type de retour

INTEGER

Exemples

L'exemple suivant compare la valeur de `3 days` à `1 year`.

```
select interval_cmp('3 days','1 year');

interval_cmp
--------------
```

```
-1
```

Cet exemple compare la valeur de `7 days` à `1 week`.

```
select interval_cmp('7 days','1 week');

interval_cmp
--------------
0
```

L'exemple suivant compare la valeur de `1 year` à `3 days`.

```
select interval_cmp('1 year','3 days');

interval_cmp
--------------
1
```

Fonction LAST_DAY

LAST_DAY renvoie la date du dernier jour du mois qui contient date. Le type de retour est toujours DATE, quel que soit le type de données de l'argument date.

Pour plus d'informations sur l'extraction de parties de dates spécifiques, consultez Fonction DATE_TRUNC.

Syntaxe

```
LAST_DAY( { date | timestamp } )
```

Arguments

date | timestamp

Colonne de type de données `DATE` ou `TIMESTAMP` ou expression implicitement évaluée à un type `DATE` ou `TIMESTAMP`.

Type de retour

DATE

Exemples

L'exemple suivant renvoie la date du dernier jour du mois en cours.

```
select last_day(sysdate);

  last_day
------------
 2014-01-31
```

L'exemple suivant renvoie le nombre de billets vendus pour chacun des 7 derniers jours du mois. Les valeurs de la colonne SALETIME sont des horodatages.

```
select datediff(day, saletime, last_day(saletime)) as "Days Remaining", sum(qtysold)
from sales
where datediff(day, saletime, last_day(saletime)) < 7
group by 1
order by 1;

days remaining |  sum
---------------+-------
             0 | 10140
             1 | 11187
             2 | 11515
             3 | 11217
             4 | 11446
             5 | 11708
             6 | 10988
(7 rows)
```

Fonction MONTHS_BETWEEN

MONTHS_BETWEEN détermine le nombre de mois entre deux dates.

Si la première date est ultérieure à la deuxième date, le résultat est positif ; Sinon, le résultat est négatif.

Si des arguments ont la valeur null, le résultat a la valeur NULL.

Syntaxe

```
MONTHS_BETWEEN( date1, date2 )
```

Arguments

date1

Colonne de type de données DATE ou expression implicitement évaluée à un type DATE.

date2

Colonne de type de données DATE ou expression implicitement évaluée à un type DATE.

Type de retour

FLOAT8

La partie du nombre entier du résultat est basée sur la différence entre les valeurs de l'année et du mois des dates. La partie fractionnée du résultat est calculée à partir des valeurs de jour et d'horodatage de la date et présume qu'un mois dure 31 jours.

Si date1 et date2 contiennent la même date dans un mois (par exemple, 15/01/14 et 15/02/14) ou le dernier jour du mois (par exemple, le 31/08/14 et le 30/09/14), le résultat est donc un nombre entier basé sur les valeurs de l'année et du mois des dates, que la partie horodatage corresponde ou non, le cas échéant.

Exemples

L'exemple suivant renvoie les mois compris entre le 18/01/1969 et le 18/03/1969.

```
select months_between('1969-01-18', '1969-03-18')
as months;

months
----------
-2
```

L'exemple suivant renvoie les mois compris entre le 18/01/1969 et le 18/01/1969.

```
select months_between('1969-01-18', '1969-01-18')
as months;

months
----------
```

```
0
```

L'exemple suivant renvoie les mois entre les premières et la dernières des projections d'un événement.

```
select eventname,
min(starttime) as first_show,
max(starttime) as last_show,
months_between(max(starttime),min(starttime)) as month_diff
from event
group by eventname
order by eventname
limit 5;

eventname         first_show             last_show              month_diff
-------------------------------------------------------------------------
.38 Special       2008-01-21 19:30:00.0  2008-12-25 15:00:00.0  11.12
3 Doors Down      2008-01-03 15:00:00.0  2008-12-01 19:30:00.0  10.94
70s Soul Jam      2008-01-16 19:30:00.0  2008-12-07 14:00:00.0  10.7
A Bronx Tale      2008-01-21 19:00:00.0  2008-12-15 15:00:00.0  10.8
A Catered Affair  2008-01-08 19:30:00.0  2008-12-19 19:00:00.0  11.35
```

Fonction NEXT_DAY

NEXT_DAY renvoie la date de la première instance d'une date spécifiée qui est ultérieure à la date donnée.

Si la valeur de day correspond au même jour de la semaine que la date donnée, la prochaine occurrence de ce jour est renvoyée.

Syntaxe

```
NEXT_DAY( { date | timestamp }, day )
```

Arguments

date | timestamp

Colonne de type de données DATE ou TIMESTAMP ou expression implicitement évaluée à un type DATE ou TIMESTAMP.

day

Chaîne contenant le nom de n’importe quel jour. La capitalisation n'a pas d'importance.

Les valeurs valides sont les suivantes.

jour	Valeurs
dimanche	D, dim, dimanche
Lundi	L, lu, lun, lundi
Mardi	mar, mardi
Mercredi	mer, mercredi
Jeudi	J, jeu, jeudi
Vendredi	V, ven, vendredi
Samedi	S, sam, samedi

Type de retour

DATE

Exemples

L’exemple suivant renvoie la date du premier mardi après le 20/08/2014.

```
select next_day('2014-08-20','Tuesday');

next_day
-----------
2014-08-26
```

L’exemple suivant renvoie la date du premier mardi après le 01/01/2008 à 5:54:44.

```
select listtime, next_day(listtime, 'Tue') from listing limit 1;

listtime            | next_day
--------------------+-----------
```

```
2008-01-01 05:54:44 | 2008-01-08
```

L'exemple suivant obtient les dates marketing cible du troisième trimestre.

```
select username, (firstname ||' '|| lastname) as name,
eventname, caldate, next_day (caldate, 'Monday') as marketing_target
from sales, date, users, event
where sales.buyerid = users.userid
and sales.eventid = event.eventid
and event.dateid = date.dateid
and date.qtr = 3
order by marketing_target, eventname, name;

username  |     name          |     eventname        |    caldate   |
 marketing_target
----------+-------------------+----------------------+--------------
+-------------------
MBO26QSG  |   Callum Atkinson | .38 Special          |  2008-07-06  | 2008-07-07
WCR50YIU  |   Erasmus Alvarez | A Doll's House       |  2008-07-03  | 2008-07-07
CKT70OIE  |   Hadassah Adkins | Ana Gabriel          |  2008-07-06  | 2008-07-07
VVG07OUO  |   Nathan Abbott   | Armando Manzanero    |  2008-07-04  | 2008-07-07
GEW77SII  |   Scarlet Avila   | August: Osage County |  2008-07-06  | 2008-07-07
ECR71CVS  |   Caryn Adkins    | Ben Folds            |  2008-07-03  | 2008-07-07
KUW82CYU  |   Kaden Aguilar   | Bette Midler         |  2008-07-01  | 2008-07-07
WZE78DJZ  |   Kay Avila       | Bette Midler         |  2008-07-01  | 2008-07-07
HXY04NVE  |   Dante Austin    | Britney Spears       |  2008-07-02  | 2008-07-07
URY81YWF  |   Wilma Anthony   | Britney Spears       |  2008-07-02  | 2008-07-07
```

Fonction SYSDATE

SYSDATE renvoie la date et l'heure actuelles selon le fuseau horaire en cours (UTC par défaut).

> **Note**
>
> SYSDATE renvoie la date et l'heure de début de la transaction en cours, pas pour le début de l'instruction en cours.

Syntaxe

```
SYSDATE
```

Cette fonction ne nécessite aucun argument.

Type de retour

TIMESTAMP

Exemples

L'exemple suivant utilise la fonction SYSDATE pour renvoyer l'horodatage complet de la date actuelle.

```
select sysdate;

timestamp
----------------------------
2008-12-04 16:10:43.976353
```

L'exemple suivant utilise la fonction SYSDATE à l'intérieur de la fonction TRUNC pour renvoyer la date du jour sans l'heure.

```
select trunc(sysdate);

trunc
------------
2008-12-04
```

La requête suivante renvoie des informations sur les ventes à des dates comprises entre la date d'émission de la requête et la date, quelle qu'elle soit, 120 jours plus tôt.

```
select salesid, pricepaid, trunc(saletime) as saletime, trunc(sysdate) as now
from sales
where saletime between trunc(sysdate)-120 and trunc(sysdate)
order by saletime asc;

 salesid | pricepaid |  saletime  |    now
---------+-----------+------------+------------
91535    |    670.00 | 2008-08-07 | 2008-12-05
91635    |    365.00 | 2008-08-07 | 2008-12-05
91901    |   1002.00 | 2008-08-07 | 2008-12-05
...
```

Fonction TIMEOFDAY

TIMEOFDAY est un alias spécial utilisé pour renvoyer le jour de la semaine, la date et l'heure comme valeur de chaîne. Cette fonction renvoie la chaîne de l'heure pour l'instruction actuelle, même lorsqu'elle se trouve dans un bloc de transaction.

Syntaxe

```
TIMEOFDAY()
```

Type de retour

VARCHAR

Exemples

L'exemple suivant renvoie la date et l'heure actuelles à l'aide de la fonction TIMEOFDAY.

```
select timeofday();

timeofday
------------
Thu Sep 19 22:53:50.333525 2013 UTC
```

Fonction TIMESTAMP_CMP

Compare la valeur de deux horodatages et renvoie un nombre entier. Si les valeurs d'horodatage sont identiques, la fonction renvoie `0`. Si la valeur du premier horodatage est supérieure, la fonction renvoie 1. Si la valeur du second horodatage est supérieure, la fonction renvoie `-1`.

Syntaxe

```
TIMESTAMP_CMP(timestamp1, timestamp2)
```

Arguments

timestamp1

Colonne de type de données `TIMESTAMP` ou expression implicitement évaluée à un type `TIMESTAMP`.

timestamp2

Colonne de type de données TIMESTAMP ou expression implicitement évaluée à un type TIMESTAMP.

Type de retour

INTEGER

Exemples

L'exemple suivant compare les horodatages et affiche les résultats de la comparaison.

```
SELECT TIMESTAMP_CMP('2008-01-24 06:43:29', '2008-01-24 06:43:29'),
 TIMESTAMP_CMP('2008-01-24 06:43:29', '2008-02-18 02:36:48'), TIMESTAMP_CMP('2008-02-18
 02:36:48', '2008-01-24 06:43:29');

timestamp_cmp  | timestamp_cmp | timestamp_cmp
---------------+---------------+---------------
             0 |            -1 |             1
```

L'exemple suivant compare les LISTTIME et SALETIME pour obtenir une liste. La valeur de TIMESTAMP_CMP est -1 pour toutes les listes, car l'horodatage de la vente est postérieur à celui de la liste.

```
select listing.listid, listing.listtime,
sales.saletime, timestamp_cmp(listing.listtime, sales.saletime)
from listing, sales
where listing.listid=sales.listid
order by 1, 2, 3, 4
limit 10;

 listid |      listtime       |      saletime       | timestamp_cmp
--------+---------------------+---------------------+---------------
      1 | 2008-01-24 06:43:29 | 2008-02-18 02:36:48 |            -1
      4 | 2008-05-24 01:18:37 | 2008-06-06 05:00:16 |            -1
      5 | 2008-05-17 02:29:11 | 2008-06-06 08:26:17 |            -1
      5 | 2008-05-17 02:29:11 | 2008-06-09 08:38:52 |            -1
      6 | 2008-08-15 02:08:13 | 2008-08-31 09:17:02 |            -1
     10 | 2008-06-17 09:44:54 | 2008-06-26 12:56:06 |            -1
     10 | 2008-06-17 09:44:54 | 2008-07-10 02:12:36 |            -1
     10 | 2008-06-17 09:44:54 | 2008-07-16 11:59:24 |            -1
```

```
     10 | 2008-06-17 09:44:54 | 2008-07-22 02:23:17 |            -1
     12 | 2008-07-25 01:45:49 | 2008-08-04 03:06:36 |            -1
(10 rows)
```

Cet exemple montre que TIMESTAMP_CMP renvoie un 0 pour des horodatages identiques :

```
select listid, timestamp_cmp(listtime, listtime)
from listing
order by 1 , 2
limit 10;

 listid | timestamp_cmp
--------+---------------
      1 |             0
      2 |             0
      3 |             0
      4 |             0
      5 |             0
      6 |             0
      7 |             0
      8 |             0
      9 |             0
     10 |             0
(10 rows)
```

Fonction TIMESTAMP_CMP_DATE

TIMESTAMP_CMP_DATE compare la valeur d'un horodatage et une date. Si les valeurs d'horodatage et de date sont identiques, la fonction renvoie `0`. Si la valeur d'horodatage est supérieure du point de vue chronologique, la fonction renvoie `1`. Si la valeur de date est supérieure, la fonction renvoie `-1`.

Syntaxe

```
TIMESTAMP_CMP_DATE(timestamp, date)
```

Arguments

timestamp

Colonne de type de données `TIMESTAMP` ou expression implicitement évaluée à un type `TIMESTAMP`.

date

Colonne de type de données DATE ou expression implicitement évaluée à un type DATE.

Type de retour

INTEGER

Exemples

L'exemple suivant compare LISTTIME à la date 2008-06-18. Les listes faites avant cette date renvoient 1 ; les listes faites après cette date renvoient -1. Les valeurs de LISTTIME sont des horodatages.

```
select listid, listtime,
timestamp_cmp_date(listtime, '2008-06-18')
from listing
order by 1, 2, 3
limit 10;


 listid |      listtime       | timestamp_cmp_date
--------+---------------------+--------------------
      1 | 2008-01-24 06:43:29 |                 -1
      2 | 2008-03-05 12:25:29 |                 -1
      3 | 2008-11-01 07:35:33 |                  1
      4 | 2008-05-24 01:18:37 |                 -1
      5 | 2008-05-17 02:29:11 |                 -1
      6 | 2008-08-15 02:08:13 |                  1
      7 | 2008-11-15 09:38:15 |                  1
      8 | 2008-11-09 05:07:30 |                  1
      9 | 2008-09-09 08:03:36 |                  1
     10 | 2008-06-17 09:44:54 |                 -1
(10 rows)
```

Fonction TIMESTAMP_CMP_TIMESTAMPTZ

TIMESTAMP_CMP_TIMESTAMPTZ compare la valeur d'une expression d'horodatage avec une expression d'horodatage avec fuseau horaire. Si les valeurs d'horodatage et d'horodatage avec fuseau horaire sont identiques, la fonction renvoie 0. Si la valeur d'horodatage est supérieure du point de vue chronologique, la fonction renvoie 1. Si la valeur d'horodatage avec fuseau horaire est supérieure, la fonction renvoie −1.

Syntaxe

```
TIMESTAMP_CMP_TIMESTAMPTZ(timestamp, timestamptz)
```

Arguments

timestamp

Colonne de type de données TIMESTAMP ou expression implicitement évaluée à un type TIMESTAMP.

timestamptz

Colonne de type de données TIMESTAMPTZ ou expression implicitement évaluée à un type TIMESTAMPTZ.

Type de retour

INTEGER

Exemples

L'exemple suivant compare les valeurs d'horodatage et d'horodatage avec fuseau horaire et affiche les résultats de la comparaison.

```
SELECT TIMESTAMP_CMP_TIMESTAMPTZ('2008-01-24 06:43:29', '2008-01-24 06:43:29+00'),
 TIMESTAMP_CMP_TIMESTAMPTZ('2008-01-24 06:43:29', '2008-02-18 02:36:48+00'),
 TIMESTAMP_CMP_TIMESTAMPTZ('2008-02-18 02:36:48', '2008-01-24 06:43:29+00');

timestamp_cmp_timestamptz  | timestamp_cmp_timestamptz | timestamp_cmp_timestamptz
---------------------------+---------------------------+--------------------------
             0             |            -1             |             1
```

Fonction TIMESTAMPTZ_CMP

TIMESTAMPTZ_CMP compare deux valeurs d'horodatage avec fuseau horaire et renvoie un nombre entier. Si les valeurs d'horodatage sont identiques, la fonction renvoie 0. Si la valeur du premier horodatage est supérieure d'un point de vue chronologique, la fonction renvoie 1. Si la valeur du second horodatage est supérieure, la fonction renvoie −1.

Syntaxe

```
TIMESTAMPTZ_CMP(timestamptz1, timestamptz2)
```

Arguments

timestamptz1

Colonne de type de données TIMESTAMPTZ ou expression implicitement évaluée à un type TIMESTAMPTZ.

timestamptz2

Colonne de type de données TIMESTAMPTZ ou expression implicitement évaluée à un type TIMESTAMPTZ.

Type de retour

INTEGER

Exemples

L'exemple suivant compare les valeurs d'horodatage avec fuseau horaire et affiche les résultats de la comparaison.

```
SELECT TIMESTAMPTZ_CMP('2008-01-24 06:43:29+00', '2008-01-24 06:43:29+00'),
 TIMESTAMPTZ_CMP('2008-01-24 06:43:29+00', '2008-02-18 02:36:48+00'),
 TIMESTAMPTZ_CMP('2008-02-18 02:36:48+00', '2008-01-24 06:43:29+00');

timestamptz_cmp  | timestamptz_cmp | timestamptz_cmp
-----------------+-----------------+----------------
        0        |       -1        |       1
```

Fonction TIMESTAMPTZ_CMP_DATE

TIMESTAMPTZ_CMP_DATE compare la valeur d'un horodatage et d'une date. Si les valeurs d'horodatage et de date sont identiques, la fonction renvoie 0. Si la valeur d'horodatage est supérieure du point de vue chronologique, la fonction renvoie 1. Si la valeur de date est supérieure, la fonction renvoie -1.

Syntaxe

```
TIMESTAMPTZ_CMP_DATE(timestamptz, date)
```

Arguments

timestamptz

Colonne de type de données TIMESTAMPTZ ou expression implicitement évaluée à un type TIMESTAMPTZ.

date

Colonne de type de données DATE ou expression implicitement évaluée à un type DATE.

Type de retour

INTEGER

Exemples

L'exemple suivant compare LISTTIME en tant qu'horodatage avec fuseau horaire à la date 2008-06-18. Les listes faites avant cette date renvoient 1 ; les listes faites après cette date renvoient -1.

```
select listid, CAST(listtime as timestamptz) as tstz,
timestamp_cmp_date(tstz, '2008-06-18')
from listing
order by 1, 2, 3
limit 10;


 listid |          tstz          | timestamptz_cmp_date
--------+------------------------+----------------------
      1 | 2008-01-24 06:43:29+00 |                   -1
      2 | 2008-03-05 12:25:29+00 |                   -1
      3 | 2008-11-01 07:35:33+00 |                    1
      4 | 2008-05-24 01:18:37+00 |                   -1
      5 | 2008-05-17 02:29:11+00 |                   -1
      6 | 2008-08-15 02:08:13+00 |                    1
      7 | 2008-11-15 09:38:15+00 |                    1
      8 | 2008-11-09 05:07:30+00 |                    1
      9 | 2008-09-09 08:03:36+00 |                    1
```

```
     10 | 2008-06-17 09:44:54+00 |              -1
(10 rows)
```

Fonction TIMESTAMPTZ_CMP_TIMESTAMP

TIMESTAMPTZ_CMP_TIMESTAMP compare la valeur d'une expression d'horodatage avec fuseau horaire à une expression d'horodatage. Si les valeurs d'horodatage avec fuseau horaire et d'horodatage sont identiques, la fonction renvoie `0`. Si la valeur d'horodatage avec fuseau horaire est supérieure du point de vue chronologique, la fonction renvoie 1. Si la valeur d'horodatage est supérieure, la fonction renvoie `-1`.

Syntaxe

```
TIMESTAMPTZ_CMP_TIMESTAMP(timestamptz, timestamp)
```

Arguments

timestamptz

Colonne de type de données `TIMESTAMPTZ` ou expression implicitement évaluée à un type `TIMESTAMPTZ`.

timestamp

Colonne de type de données `TIMESTAMP` ou expression implicitement évaluée à un type `TIMESTAMP`.

Type de retour

INTEGER

Exemples

L'exemple suivant compare les valeurs d'horodatage avec fuseau horaire et d'horodatage et affiche les résultats de la comparaison.

```
SELECT TIMESTAMPTZ_CMP_TIMESTAMP('2008-01-24 06:43:29+00', '2008-01-24 06:43:29'),
 TIMESTAMPTZ_CMP_TIMESTAMP('2008-01-24 06:43:29+00', '2008-02-18 02:36:48'),
 TIMESTAMPTZ_CMP_TIMESTAMP('2008-02-18 02:36:48+00', '2008-01-24 06:43:29');

timestamptz_cmp_timestamp  | timestamptz_cmp_timestamp | timestamptz_cmp_timestamp
---------------------------+---------------------------+--------------------------
```

```
        0       |       -1        |        1
```

Fonction TIMEZONE

TIMEZONE renvoie un horodatage pour la valeur du fuseau horaire et de l'horodatage spécifiée.

Pour plus d'informations et d'exemples sur la façon de définir un fuseau horaire, consultez timezone.

Pour plus d'informations et d'exemples sur la façon de convertir un fuseau horaire, consultez CONVERT_TIMEZONE.

Syntaxe

```
TIMEZONE('timezone', { timestamp | timestamptz })
```

Arguments

timezone

Le fuseau horaire de la valeur de retour. Le fuseau horaire peut être spécifié comme nom de fuseau horaire (tel que **'Africa/Kampala'** ou **'Singapore'**) ou comme abréviation de fuseau horaire (telle que **'UTC'** ou **'PDT'**). Pour afficher la liste des noms de fuseaux horaires pris en charge, exécutez la commande suivante.

```
select pg_timezone_names();
```

Pour afficher la liste des abréviations de fuseaux horaires prises en charge, exécutez la commande suivante.

```
select pg_timezone_abbrevs();
```

Pour plus d'informations et d'exemples, consultez Remarques sur l'utilisation de fuseaux horaires.

timestamp | timestamptz

Expression qui a pour résultat un type TIMESTAMP ou TIMESTAMPTZ, ou une valeur qui peut être implicitement convertie en horodatage ou horodatage avec fuseau horaire.

Type de retour

TIMESTAMPTZ lorsqu'il est utilisé avec une expression TIMESTAMP.

TIMESTAMP lorsqu'il est utilisé avec une expression TIMESTAMPTZ.

Exemples

L'exemple suivant renvoie un horodatage pour le fuseau horaire UTC en utilisant l'horodatage `2008-06-17 09:44:54` du fuseau horaire PST.

```
SELECT TIMEZONE('PST', '2008-06-17 09:44:54');

timezone
-----------------------
2008-06-17 17:44:54+00
```

L'exemple suivant renvoie un horodatage pour le fuseau horaire PST en utilisant l'horodatage avec le fuseau horaire UTC `2008-06-17 09:44:54+00`.

```
SELECT TIMEZONE('PST', timestamptz('2008-06-17 09:44:54+00'));

timezone
-----------------------
2008-06-17 01:44:54
```

Fonction TO_TIMESTAMP

TO_TIMESTAMP convertit une chaîne TIMESTAMP en TIMESTAMPTZ. Pour obtenir une liste des fonctions de date et d'heure supplémentaires pour Amazon Redshift, consultez Fonctions de date et d'heure.

Syntaxe

```
to_timestamp(timestamp, format)
```

```
to_timestamp (timestamp, format, is_strict)
```

Arguments

timestamp

Chaîne qui représente une valeur d'horodatage au format spécifié par format. Si cet argument est laissé vide, la valeur de l'horodatage est fixée par défaut à `0001-01-01 00:00:00`.

format

Valeur de chaine littérale qui définit le format de la valeur timestamp. Formats qui incluent un fuseau horaire (**TZ**, **tz** ou **OF**) ne sont pas pris en charge comme entrée. Pour les formats d'horodatage valides, consultez Chaînes de format datetime.

is_strict

Valeur booléenne facultative qui spécifie si une erreur est renvoyée lorsqu'une valeur timestamp en entrée est hors de portée. Quand is_strict est défini sur TRUE, une erreur est renvoyée s'il y a une valeur hors de portée. Quand is_strict est défini sur FALSE, qui est la valeur par défaut, les valeurs en dépassement sont acceptées.

Type de retour

TIMESTAMPTZ

Exemples

L'exemple suivant montre l'utilisation de la fonction TO_TIMESTAMP pour convertir une chaîne TIMESTAMP en une chaîne TIMESTAMPTZ.

```
select sysdate, to_timestamp(sysdate, 'YYYY-MM-DD HH24:MI:SS') as second;

timestamp                  | second
---------------------------   ----------------------
2021-04-05 19:27:53.281812 | 2021-04-05 19:27:53+00
```

Il est possible de transmettre la partie TO_TIMESTAMP d'une date. Les autres parties de la date sont définies sur des valeurs par défaut. L'heure est incluse dans la sortie :

```
SELECT TO_TIMESTAMP('2017','YYYY');

to_timestamp
---------------------------
2017-01-01 00:00:00+00
```

L'instruction SQL suivante convertit la chaîne « 2011-12-18 24:38:15 » en TIMESTAMPTZ. Le résultat est une valeur TIMESTAMPTZ qui tombe le jour suivant, car le nombre d'heures est supérieur à 24 heures :

```
SELECT TO_TIMESTAMP('2011-12-18 24:38:15', 'YYYY-MM-DD HH24:MI:SS');

to_timestamp
---------------------
2011-12-19 00:38:15+00
```

L'instruction SQL suivante convertit la chaîne « 2011-12-18 24:38:15 » en TIMESTAMPTZ. Le résultat est une erreur, car la valeur time dans l'horodatage est supérieure à 24 heures.

```
SELECT TO_TIMESTAMP('2011-12-18 24:38:15', 'YYYY-MM-DD HH24:MI:SS', TRUE);

ERROR:  date/time field time value out of range: 24:38:15.0
```

Fonction TRUNC

Tronque une valeur `TIMESTAMP` et renvoie une valeur `DATE`.

Cette fonction peut également tronquer un nombre. Pour de plus amples informations, veuillez consulter Fonction TRUNC.

Syntaxe

```
TRUNC(timestamp)
```

Arguments

timestamp

Colonne de type de données `TIMESTAMP` ou expression implicitement évaluée à un type `TIMESTAMP`.

Pour renvoyer une valeur d'horodatage avec `00:00:00` comme heure, convertissez le résultat de la fonction en `TIMESTAMP`.

Type de retour

DATE

Exemples

L'exemple suivant renvoie la partie de date du résultat de la fonction SYSDATE (qui renvoie un horodatage).

```
SELECT SYSDATE;

+----------------------------+
|         timestamp          |
+----------------------------+
| 2011-07-21 10:32:38.248109 |
+----------------------------+

SELECT TRUNC(SYSDATE);

+------------+
|   trunc    |
+------------+
| 2011-07-21 |
+------------+
```

L'exemple suivant applique la fonction TRUNC à une colonne `TIMESTAMP`. Le type de retour est une date.

```
SELECT TRUNC(starttime) FROM event
ORDER BY eventid LIMIT 1;

+------------+
|   trunc    |
+------------+
| 2008-01-25 |
+------------+
```

L'exemple suivant renvoie une valeur d'horodatage avec `00:00:00` comme heure en convertissant le résultat de la fonction TRUNC en `TIMESTAMP`.

```
SELECT CAST((TRUNC(SYSDATE)) AS TIMESTAMP);

+---------------------+
|        trunc        |
+---------------------+
```

```
| 2011-07-21 00:00:00 |
+---------------------+
```

Parties de date pour les fonctions de date ou d'horodatage

Le tableau suivant identifie les noms de partie de date et d'horodatage et les abréviations qui sont acceptées comme arguments pour les fonctions suivantes :

- DATEADD
- DATEDIFF
- DATE_PART
- EXTRACT

Partie de date ou de temps	Abréviations
millénaire, millénaires	mil
siècle, siècles	s, siècle, siècles
décennie, décennies	déc
époque	époque (prise en charge par la EXTRACT)
année, années	an, ans
trimestre, trimestres	trim
mois	mois
semaine, semaines	s, sem
jour de la semaine	jdls (pris en charge par les fonctions DATE_PART et Fonction EXTRACT) Renvoie un nombre entier compris entre 0 et 6, en commençant par le dimanche.

Partie de date ou de temps	Abréviations
	Note La partie de date DOW se comporte différemment de la partie de date jour de la semaine (D) utilisée pour les chaînes au format datetime. D s'appuie sur des nombres entiers compris entre 1 et 7, où le dimanche est 1. Pour plus d'informations, consultez Chaînes de format datetime.
jour de l'année	dayofyear, doy, dy, yearday (prise en charge par la EXTRACT)
jour, jours	d
heure, heures	h
minute, minutes	m, min
seconde, secondes	s
milliseconde, millisecondes	ms
microseconde, microsecondes	µs
timezone, timezone_hour, timezone_minute	Pris en charge par la EXTRACT pour l'horodatage avec fuseau horaire (TIMESTAMPTZ) uniquement.

Variations de résultats avec les secondes, les millisecondes et les microsecondes

Des différences mineures dans les résultats de la requête se produisent lorsque d'autres fonctions de date spécifient les secondes, les millisecondes ou les microsecondes comme des parties de date :

- La fonction EXTRACT renvoie des nombres entiers pour la partie de date spécifiée uniquement, sans tenir compte des parties de date de niveau supérieur et inférieur. Si la partie de date spécifiée est les secondes, les millisecondes et les microsecondes ne figurent pas dans le résultat. Si la partie de date spécifiée est les millisecondes, les secondes et les microsecondes ne sont pas

incluses. Si la partie de date spécifiée est les microsecondes, les secondes et les millisecondes ne sont pas incluses.

- La fonction DATE_PART renvoie la seconde partie complète de l'horodatage, quelle que soit la partie de date spécifiée, en renvoyant une valeur décimale ou un nombre entier comme requis.

Par exemple, comparez les résultats des requêtes suivantes :

```
create table seconds(micro timestamp);

insert into seconds values('2009-09-21 11:10:03.189717');

select extract(sec from micro) from seconds;

date_part
-----------
3

select date_part(sec, micro) from seconds;

pgdate_part
-------------
3.189717
```

Remarques sur CENTURY, EPOCH, DECADE et MIL

CENTURY ou CENTURIES

Amazon Redshift interprète un CENTURY comme démarrant avec l'année ###1 et se terminant par l'année ###0 :

```
select extract (century from timestamp '2000-12-16 12:21:13');
date_part
-----------
20

select extract (century from timestamp '2001-12-16 12:21:13');
date_part
-----------
21
```

EPOCH

L'implémentation d'Amazon Redshift d'EPOCH est associée à 1970-01-01 00:00:00.000000 quel que soit le fuseau horaire sur lequel réside le cluster. Vous devrez peut-être décaler les résultats de la différence en heures selon le fuseau horaire sur lequel se trouve le cluster.

L'exemple suivant illustre les éléments suivants :

1. Crée une table appelée EVENT_EXAMPLE en fonction de la table EVENT. Cette commande CREATE AS utilise la fonction DATE_PART pour créer une colonne de date (appelée PGDATE_PART par défaut) pour stocker la valeur epoch de chaque événement.
2. Sélectionne le type de colonne et de données d'EVENT_EXAMPLE de PG_TABLE_DEF.
3. Sélectionne EVENTNAME, STARTTIME et PGDATE_PART de la table EVENT_EXAMPLE pour afficher les différents formats de date et d'heure.
4. Sélectionne EVENTNAME et STARTTIME à partir de EVENT EXAMPLE tel quel. Convertit les valeurs d'époque dans PGDATE_PART à l'aide d'un intervalle de 1 seconde dans un horodatage sans fuseau horaire et renvoie les résultats dans une colonne appelée CONVERTED_TIMESTAMP.

```
create table event_example
as select eventname, starttime, date_part(epoch, starttime) from event;

select "column", type from pg_table_def where tablename='event_example';

     column    |            type
---------------+-----------------------------
 eventname     | character varying(200)
 starttime     | timestamp without time zone
 pgdate_part   | double precision
(3 rows)
```

```
select eventname, starttime, pgdate_part from event_example;

   eventname          |      starttime      | pgdate_part
----------------------+---------------------+-------------
 Mamma Mia!           | 2008-01-01 20:00:00 |  1199217600
 Spring Awakening     | 2008-01-01 15:00:00 |  1199199600
 Nas                  | 2008-01-01 14:30:00 |  1199197800
 Hannah Montana       | 2008-01-01 19:30:00 |  1199215800
 K.D. Lang            | 2008-01-01 15:00:00 |  1199199600
 Spamalot             | 2008-01-02 20:00:00 |  1199304000
```

```
 Macbeth              | 2008-01-02 15:00:00 |  1199286000
 The Cherry Orchard   | 2008-01-02 14:30:00 |  1199284200
 Macbeth              | 2008-01-02 19:30:00 |  1199302200
 Demi Lovato          | 2008-01-02 19:30:00 |  1199302200


select eventname,
starttime,
timestamp with time zone 'epoch' + pgdate_part * interval '1 second' AS
 converted_timestamp
from event_example;

       eventname      |      starttime      | converted_timestamp
----------------------+---------------------+---------------------
 Mamma Mia!           | 2008-01-01 20:00:00 | 2008-01-01 20:00:00
 Spring Awakening     | 2008-01-01 15:00:00 | 2008-01-01 15:00:00
 Nas                  | 2008-01-01 14:30:00 | 2008-01-01 14:30:00
 Hannah Montana       | 2008-01-01 19:30:00 | 2008-01-01 19:30:00
 K.D. Lang            | 2008-01-01 15:00:00 | 2008-01-01 15:00:00
 Spamalot             | 2008-01-02 20:00:00 | 2008-01-02 20:00:00
 Macbeth              | 2008-01-02 15:00:00 | 2008-01-02 15:00:00
 The Cherry Orchard   | 2008-01-02 14:30:00 | 2008-01-02 14:30:00
 Macbeth              | 2008-01-02 19:30:00 | 2008-01-02 19:30:00
 Demi Lovato          | 2008-01-02 19:30:00 | 2008-01-02 19:30:00
 ...
```

DECADE ou DECADES

Amazon Redshift interprète DECADE ou DECADES DATEPART en fonction du calendrier courant. Par exemple, si le calendrier commun commence à partir de l'année 1, la première décennie (décennie 1) est 0001-01-01 jusqu'au 0009-12-31, et la deuxième décennie (décennie 2) du 0010-01-01 au 0019-12-31. Par exemple, la décennie 201 s'étend du 2000-01-01 au 2009-12-31 :

```
select extract(decade from timestamp '1999-02-16 20:38:40');
date_part
-----------
200

select extract(decade from timestamp '2000-02-16 20:38:40');
date_part
-----------
201
```

```
select extract(decade from timestamp '2010-02-16 20:38:40');
date_part
-----------
202
```

MIL ou MILS

Amazon Redshift interprète un MIL comme démarrant avec le premier jour de l'année #001 et se terminant par le dernier jour de l'année #000 :

```
select extract (mil from timestamp '2000-12-16 12:21:13');
date_part
-----------
2

select extract (mil from timestamp '2001-12-16 12:21:13');
date_part
-----------
3
```

Fonctions de hachage

Rubriques

- Fonction CHECKSUM
- Fonction farmFingerprint64
- Fonction FUNC_ SHA1
- Fonction FNV_HASH
- MD5 fonction
- Fonction SHA
- SHA1 fonction
- SHA2 fonction
- MURMUR3_32 HASH

Une fonction de hachage est une fonction mathématique qui convertit une valeur d'entrée numérique en une autre valeur.

Fonction CHECKSUM

Calcule une valeur checksum pour créer un index de hachage.

Syntaxe

```
CHECKSUM(expression)
```

Argument

expression

L'expression d'entrée doit être un type de données VARCHAR, INTEGER ou DECIMAL.

Type de retour

La fonction CHECKSUM renvoie un nombre entier.

exemple

L'exemple suivant calcule une valeur checksum de la colonne COMMISSION :

```
select checksum(commission)
from sales
order by salesid
limit 10;

checksum
----------
10920
1140
5250
2625
2310
5910
11820
2955
8865
975
(10 rows)
```

Fonction farmFingerprint64

Calcule la valeur farmhash de l'argument en entrée à l'aide de la fonction `Fingerprint64`.

Syntaxe

```
farmFingerprint64(expression)
```

Argument

expression

L'expression en entrée doit être un type de données `VARCHAR` ou `VARBYTE`.

Type de retour

La fonction `farmFingerprint64` renvoie un `BIGINT`.

exemple

L'exemple suivant renvoie la valeur `farmFingerprint64` de `Amazon Redshift` qui est entré en tant que type de données `VARCHAR`.

```
SELECT farmFingerprint64('Amazon Redshift');
```

```
  farmfingerprint64
---------------------
 8085098817162212970
```

L'exemple suivant renvoie la valeur `farmFingerprint64` de `Amazon Redshift` qui est entré en tant que type de données `VARBYTE`.

```
SELECT farmFingerprint64('Amazon Redshift'::varbyte);
```

```
  farmfingerprint64
---------------------
 8085098817162212970
```

Fonction FUNC_ SHA1

Synonyme de SHA1 fonction.

Consultez SHA1 fonction.

Fonction FNV_HASH

Calcule la fonction de hachage non cryptographique FNV-1a 64 bits pour tous les types de données de base.

Syntaxe

```
FNV_HASH(value [, seed])
```

Arguments

valeur

Valeur d'entrée à hacher. Amazon Redshift utilise la représentation binaire de la valeur pour hacher la valeur d'entrée ; par exemple, les valeurs INTEGER sont hachées en utilisant 4 octets et les valeurs BIGINT sont hachées en utilisant 8 octets. En outre, le hachage des entrées CHAR et VARCHAR n'ignore pas les espaces de fin.

Seed (Noyau)

La graine BIGINT de la fonction de hachage est facultative. Si ce n'est pas indiqué, Amazon Redshift utilise le noyau FNV par défaut. Cela permet de combiner le hachage de plusieurs colonnes sans conversions ni concaténations.

Type de retour

BIGINT

exemple

Les exemples suivants renvoient le hachage FNV d'un nombre, la chaîne Amazon Redshift et la concaténation des deux.

```
select fnv_hash(1);
        fnv_hash
```

```
----------------------
 -5968735742475085980
(1 row)
```

```
select fnv_hash('Amazon Redshift');
      fnv_hash
---------------------
 7783490368944507294
(1 row)
```

```
select fnv_hash('Amazon Redshift', fnv_hash(1));
       fnv_hash
----------------------
 -2202602717770968555
(1 row)
```

Notes d'utilisation

- Pour calculer le hachage d'une table avec plusieurs colonnes, vous pouvez calculer le hachage FNV de la première colonne et le transmettre en tant que noyau au hachage de la deuxième colonne. Ensuite, il passe le hachage FNV de la deuxième colonne en tant que noyau au hachage de la troisième colonne.

  L'exemple suivant crée des noyaux pour hacher une table comportant plusieurs colonnes.

  ```
  select fnv_hash(column_3, fnv_hash(column_2, fnv_hash(column_1))) from sample_table;
  ```

- La même propriété peut être utilisée pour calculer le hachage d'une concaténation de chaînes.

  ```
  select fnv_hash('abcd');
         fnv_hash
  ---------------------
   -281581062704388899
  (1 row)
  ```

  ```
  select fnv_hash('cd', fnv_hash('ab'));
        fnv_hash
  ---------------------
   -281581062704388899
  (1 row)
  ```

- La fonction de hachage utilise le type de l'entrée pour déterminer le nombre d'octets à hacher. Utilisez la conversion de types pour appliquer un type spécifique, si nécessaire.

  Les exemples suivants utilisent différents types d'entrée pour produire des résultats différents.

```
select fnv_hash(1::smallint);
      fnv_hash
--------------------
 589727492704079044
(1 row)
```

```
select fnv_hash(1);
       fnv_hash
----------------------
 -5968735742475085980
(1 row)
```

```
select fnv_hash(1::bigint);
       fnv_hash
----------------------
 -8517097267634966620
(1 row)
```

MD5 fonction

Utilise la fonction de hachage MD5 cryptographique pour convertir une chaîne de longueur variable en une chaîne de 32 caractères qui est une représentation textuelle de la valeur hexadécimale d'une somme de contrôle de 128 bits.

Syntaxe

```
MD5(string)
```

Arguments

string

Chaîne de longueur variable.

Type de retour

La MD5 fonction renvoie une chaîne de 32 caractères qui est une représentation textuelle de la valeur hexadécimale d'une somme de contrôle de 128 bits.

Exemples

L'exemple suivant illustre la valeur de 128 bits de la chaîne « Amazon Redshift » :

```
select md5('Amazon Redshift');
md5
----------------------------------
f7415e33f972c03abd4f3fed36748f7a
(1 row)
```

Fonction SHA

Synonyme de SHA1 fonction.

Consultez SHA1 fonction.

SHA1 fonction

La SHA1 fonction utilise la fonction de hachage SHA1 cryptographique pour convertir une chaîne de longueur variable en une chaîne de 40 caractères qui est une représentation textuelle de la valeur hexadécimale d'une somme de contrôle de 160 bits.

Syntaxe

SHA1 est un synonyme de Fonction SHA et. Fonction FUNC_ SHA1

```
SHA1(string)
```

Arguments

string

Chaîne de longueur variable.

Type de retour

La SHA1 fonction renvoie une chaîne de 40 caractères qui est une représentation textuelle de la valeur hexadécimale d'une somme de contrôle de 160 bits.

exemple

L'exemple suivant renvoie la valeur de 160 bits du mot « Amazon Redshift » :

```
select sha1('Amazon Redshift');
```

SHA2 fonction

La SHA2 fonction utilise la fonction de hachage SHA2 cryptographique pour convertir une chaîne de longueur variable en chaîne de caractères. La chaîne de caractères est une représentation textuelle de la valeur hexadécimale du total de contrôle avec le nombre de bits spécifié.

Syntaxe

```
SHA2(string, bits)
```

Arguments

string

Chaîne de longueur variable.

integer

Nombre de bits dans les fonctions de hachage. Les valeurs valides sont 0 (identique à 256), 224, 256, 384 et 512.

Type de retour

La SHA2 fonction renvoie une chaîne de caractères qui est une représentation textuelle de la valeur hexadécimale de la somme de contrôle ou une chaîne vide si le nombre de bits n'est pas valide.

exemple

L'exemple suivant renvoie la valeur de 256 bits du mot « Amazon Redshift » :

```
select sha2('Amazon Redshift', 256);
```

MURMUR3_32 HASH

La fonction MURMUR3 _32_HASH calcule le hachage non cryptographique Murmur3A 32 bits pour tous les types de données courants, y compris les types numériques et de chaîne.

Syntaxe

```
MURMUR3_32_HASH(value [, seed])
```

Arguments

valeur

Valeur d'entrée à hacher. Amazon Redshift hache la représentation binaire de la valeur d'entrée. Ce comportement est similaire à Fonction FNV_HASH, mais la valeur est convertie en la représentation binaire spécifiée par la spécification de hachage Murmur3 de 32 bits d'Apache Iceberg.

Seed (Noyau)

Le noyau INT de la fonction de hachage. Cet argument est facultatif. Si ce n'est pas indiqué, Amazon Redshift utilise le noyau 0 par défaut. Cela permet de combiner le hachage de plusieurs colonnes sans conversions ni concaténations.

Type de retour

La fonction renvoie un INT.

exemple

Les exemples suivants renvoient le hachage Murmur3 d'un nombre, la chaîne Amazon Redshift et la concaténation des deux.

```
select MURMUR3_32_HASH(1);

    MURMUR3_32_HASH
----------------------
 1392991556
(1 row)
```

```
select MURMUR3_32_HASH('Amazon Redshift');

    MURMUR3_32_HASH
----------------------
 -1563580564
(1 row)
```

```
select MURMUR3_32_HASH('Amazon Redshift', MURMUR3_32_HASH(1));

    MURMUR3_32_HASH
----------------------
 -1346554171
(1 row)
```

Notes d'utilisation

Pour calculer le hachage d'une table avec plusieurs colonnes, vous pouvez calculer le hachage Murmur3 de la première colonne et le transmettre en tant que noyau au hachage de la deuxième colonne. Ensuite, il passe le hachage Murmur3 de la deuxième colonne en tant que noyau au hachage de la troisième colonne.

L'exemple suivant crée des noyaux pour hacher une table comportant plusieurs colonnes.

```
select MURMUR3_32_HASH(column_3, MURMUR3_32_HASH(column_2, MURMUR3_32_HASH(column_1)))
 from sample_table;
```

La même propriété peut être utilisée pour calculer le hachage d'une concaténation de chaînes.

```
select MURMUR3_32_HASH('abcd');

   MURMUR3_32_HASH
----------------------
 1139631978
(1 row)
```

```
select MURMUR3_32_HASH('cd', MURMUR3_32_HASH('ab'));

   MURMUR3_32_HASH
----------------------
 1711522338
```

```
(1 row)
```

La fonction de hachage utilise le type de l'entrée pour déterminer le nombre d'octets à hacher. Utilisez la conversion de types pour appliquer un type spécifique, si nécessaire.

Les exemples suivants utilisent différents types d'entrée pour produire des résultats différents.

```
select MURMUR3_32_HASH(1, MURMUR3_32_HASH(1));

   MURMUR3_32_HASH
--------------------
 -1193428387
(1 row)
```

```
select MURMUR3_32_HASH(1);

   MURMUR3_32_HASH
----------------------
 1392991556
(1 row)
```

```
select MURMUR3_32_HASH(1, MURMUR3_32_HASH(2));

   MURMUR3_32_HASH
----------------------
 1179621905
(1 row)
```

HyperLogLog fonctions

Vous trouverez ci-dessous les descriptions des HyperLogLog fonctions SQL prises en charge par Amazon Redshift.

Rubriques

- Fonction HLL
- Fonction HLL_CREATE_SKETCH
- Fonction HLL_CARDINALITY
- Fonction HLL_COMBINE
- Fonction HLL_COMBINE_SKETCHES

Fonction HLL

La fonction HLL renvoie la HyperLogLog cardinalité des valeurs d'expression d'entrée. La fonction HLL fonctionne avec tous les types de données sauf le type de données HLLSKETCH. La fonction HLL ignore les valeurs NULL. Lorsqu'il n'y a pas de lignes dans une table ou que toutes les lignes sont NULL, la cardinalité résultante est 0.

Syntaxe

```
HLL (aggregate_expression)
```

Argument

aggregate_expression

Toute expression valide qui fournit la valeur à un agrégat, telle qu'un nom de colonne. Cette fonction prend en charge n'importe quel type de données d'entrée, sauf HLLSKETCH, GEOMETRY, GEOGRAPHY et VARBYTE.

Type de retour

La fonction HLL renvoie un BIGINT ou INT8 une valeur.

Exemples

L'exemple suivant renvoie la cardinalité de la colonne `an_int` de la table `a_table`.

```
CREATE TABLE a_table(an_int INT);
INSERT INTO a_table VALUES (1), (2), (3), (4);

SELECT hll(an_int) AS cardinality FROM a_table;
cardinality
-------------
4
```

Fonction HLL_CREATE_SKETCH

La fonction HLL_CREATE_SKETCH renvoie un type de données HLLSKETCH qui encapsule les valeurs d'expression en entrée. La fonction HLL_CREATE_SKETCH fonctionne avec n'importe

quel type de données et ignore les valeurs NULL. Lorsqu'il n'y a pas de lignes dans une table ou que toutes les lignes sont NULL, le schéma résultant n'a pas de paires index-valeur telles que `{"version":1,"logm":15,"sparse":{"indices":[],"values":[]}}`.

Syntaxe

```
HLL_CREATE_SKETCH (aggregate_expression)
```

Argument

aggregate_expression

Toute expression valide qui fournit la valeur à un agrégat, telle qu'un nom de colonne. Les valeurs NULL sont ignorées. Cette fonction prend en charge n'importe quel type de données d'entrée, sauf HLLSKETCH, GEOMETRY, GEOGRAPHY et VARBYTE.

Type de retour

La fonction HLL_CREATE_SKETCH renvoie une valeur HLLSKETCH.

Exemples

L'exemple suivant renvoie le type HLLSKETCH pour la colonne `an_int` de la table `a_table`. Un objet JSON est utilisé pour représenter une HyperLogLog esquisse éparse lors de l'importation, de l'exportation ou de l'impression d'esquisses. Une représentation sous forme de chaîne (au format Base64) est utilisée pour représenter une HyperLogLog esquisse dense.

```
CREATE TABLE a_table(an_int INT);
INSERT INTO a_table VALUES (1), (2), (3), (4);

SELECT hll_create_sketch(an_int) AS sketch FROM a_table;
sketch
-----------------------------------------------------------------------------------------------
{"version":1,"logm":15,"sparse":{"indices":
[20812342,20850007,22362299,47158030],"values":[1,2,1,1]}}
(1 row)
```

Fonction HLL_CARDINALITY

La fonction HLL_CARDINALITY renvoie la cardinalité du type de données HLLSKETCH en entrée.

Syntaxe

```
HLL_CARDINALITY (hllsketch_expression)
```

Argument

hllsketch_expression

Expression valide qui correspond à un type HLLSKETCH, par exemple un nom de colonne. La valeur en entrée est le type de données HLLSKETCH.

Type de retour

La fonction HLL_CARDINALITY renvoie un BIGINT ou une valeur. INT8

Exemples

L'exemple suivant renvoie la cardinalité de la colonne `sketch` de la table `hll_table`.

```
CREATE TABLE a_table(an_int INT, b_int INT);
INSERT INTO a_table VALUES (1,1), (2,1), (3,1), (4,1), (1,2), (2,2), (3,2), (4,2),
 (5,2), (6,2);

CREATE TABLE hll_table (sketch HLLSKETCH);
INSERT INTO hll_table select hll_create_sketch(an_int) from a_table group by b_int;

SELECT hll_cardinality(sketch) AS cardinality FROM hll_table;
cardinality
-------------
6
4
(2 rows)
```

Fonction HLL_COMBINE

La fonction d'agrégation HLL_COMBINE renvoie un type de données HLLSKETCH qui combine toutes les valeurs HLLSKETCH en entrée.

La combinaison de deux HyperLogLog esquisses ou plus constitue un nouveau HLLSKETCH qui encapsule les informations relatives à l'union des valeurs distinctes représentées par chaque esquisse en entrée. Après avoir combiné les schémas, Amazon Redshift extrait la cardinalité de

l'union de deux ou plusieurs jeux de données. Pour plus d'informations sur la façon de combiner plusieurs schémas, consultez Exemple : renvoyer une HyperLogLog esquisse en combinant plusieurs esquisses.

Syntaxe

```
HLL_COMBINE (hllsketch_expression)
```

Argument

hllsketch_expression

Expression valide qui correspond à un type HLLSKETCH, par exemple un nom de colonne. La valeur en entrée est le type de données HLLSKETCH.

Type de retour

La fonction HLL_COMBINE renvoie un type HLLSKETCH.

Exemples

L'exemple suivant renvoie les valeurs HLLSLETCH combinées dans la table `hll_table`.

```
CREATE TABLE a_table(an_int INT, b_int INT);
INSERT INTO a_table VALUES (1,1), (2,1), (3,1), (4,1), (1,2), (2,2), (3,2), (4,2),
 (5,2), (6,2);

CREATE TABLE hll_table (sketch HLLSKETCH);
INSERT INTO hll_table select hll_create_sketch(an_int) from a_table group by b_int;

SELECT hll_combine(sketch) AS sketches FROM hll_table;
sketches
--------------------------------------------------------------------------------------------
{"version":1,"logm":15,"sparse":{"indices":
[20812342,20850007,22362299,40314817,42650774,47158030],"values":[1,2,1,3,2,1]}}
(1 row)
```

Fonction HLL_COMBINE_SKETCHES

HLL_COMBINE_SKETCHES est une fonction scalaire qui prend comme entrée deux valeurs HLLSKETCH et les combine en une seule HLLSKETCH.

La combinaison de deux HyperLogLog esquisses ou plus constitue un nouveau HLLSKETCH qui encapsule les informations relatives à l'union des valeurs distinctes représentées par chaque esquisse en entrée.

Syntaxe

```
HLL_COMBINE_SKETCHES (hllsketch_expression1, hllsketch_expression2)
```

Argument

hllsketch_expression1 et hllsketch_expression2

Expression valide qui correspond à un type HLLSKETCH, par exemple un nom de colonne.

Type de retour

La fonction HLL_COMBINE_SKETCHES renvoie un type HLLSKETCH.

Exemples

L'exemple suivant renvoie les valeurs HLLSLETCH combinées dans la table `hll_table`.

```
WITH tbl1(x, y)
     AS (SELECT Hll_create_sketch(1),
                Hll_create_sketch(2)
         UNION ALL
         SELECT Hll_create_sketch(3),
                Hll_create_sketch(4)
         UNION ALL
         SELECT Hll_create_sketch(5),
                Hll_create_sketch(6)
         UNION ALL
         SELECT Hll_create_sketch(7),
                Hll_create_sketch(8)),
     tbl2(x, y)
     AS (SELECT Hll_create_sketch(9),
                Hll_create_sketch(10)
         UNION ALL
         SELECT Hll_create_sketch(11),
                Hll_create_sketch(12)
         UNION ALL
```

```
        SELECT Hll_create_sketch(13),
               Hll_create_sketch(14)
        UNION ALL
        SELECT Hll_create_sketch(15),
               Hll_create_sketch(16)
        UNION ALL
        SELECT Hll_create_sketch(NULL),
               Hll_create_sketch(NULL)),
     tbl3(x, y)
     AS (SELECT *
         FROM   tbl1
         UNION ALL
         SELECT *
         FROM   tbl2)
SELECT Hll_combine_sketches(x, y)
FROM   tbl3;
```

Fonctions JSON

Rubriques

- Fonction JSON_PARSE
- Fonction CAN_JSON_PARSE
- Fonction JSON_SERIALIZE
- Fonction JSON_SERIALIZE_TO_VARBYTE
- Fonctions JSON basées sur du texte

> **Note**
>
> Nous vous recommandons d'utiliser les fonctions suivantes pour travailler avec JSON :
>
> - Fonction JSON_PARSE
> - Fonction CAN_JSON_PARSE
> - Fonction JSON_SERIALIZE
> - Fonction JSON_SERIALIZE_TO_VARBYTE
>
> Avec JSON_PARSE, il suffit de convertir le texte JSON en valeur de type SUPER une seule fois lors de l'ingestion, après quoi vous pouvez opérer sur les valeurs SUPER. Amazon

Redshift analyse les valeurs SUPER plus efficacement que VARCHAR, qui est la sortie des fonctions JSON basées sur du texte. Pour plus d'informations sur l'utilisation du type de données SUPER, consultezDonnées semi-structurées dans Amazon Redshift.

Lorsque vous avez besoin de stocker un ensemble relativement petit de paires clé-valeur, vous pouvez économiser de l'espace en stockant les données au format JSON. Étant donné que les chaînes au format JSON peuvent être stockées dans une seule colonne, l'utilisation de JSON peut être plus efficace que de stocker vos données sous forme de table. Par exemple, supposons que vous ayez une table partiellement remplie, dans laquelle vous avez besoin de nombreuses colonnes pour représenter entièrement tous les attributs possibles, mais que la plupart des valeurs de colonnes ont une valeur NULL pour n'importe quelle ligne ou colonne donnée. En utilisant JSON pour le stockage, vous pourriez stocker les données pour une ligne des paires clé:valeur dans une seule chaîne JSON et éliminer les colonnes de la table partiellement renseignées.

En outre, vous pouvez facilement modifier les chaînes JSON pour stocker des paires clé:valeur supplémentaires lorsque votre schéma JSON change sans avoir à ajouter de colonnes à une table.

Nous vous conseillons d'utiliser JSON avec modération. JSON n'est pas le bon choix pour stocker des ensembles de données plus volumineux parce que, en stockant des données disparates dans une seule colonne, JSON n'exploite pas l'architecture de stockage de colonnes d'Amazon Redshift. Bien qu'Amazon Redshift prenne en charge les fonctions JSON sur les colonnes CHAR et VARCHAR, nous vous recommandons d'utiliser SUPER pour le traitement des données au format de sérialisation JSON. SUPER utilise une représentation sans schéma post-analyse qui peut interroger efficacement les données hiérarchiques. Pour plus d'informations sur le type de données SUPER, consultez Données semi-structurées dans Amazon Redshift.

JSON utilise des chaînes de texte codées UTF-8, les chaînes JSON peuvent donc être stockées sous forme de types de données CHAR ou VARCHAR.

Les chaînes JSON doivent être au bon format JSON, selon les règles suivantes :

- Le JSON de niveau racine peut être un objet JSON ou un tableau JSON. Un objet JSON est un ensemble non trié de paires clé:valeur séparées par des virgules délimitées par des accolades.

  Par exemple, `{"one":1, "two":2}`
- Un tableau JSON est un ensemble ordonné de valeurs séparées par des virgules délimitées par des crochets.

Voici un exemple : `["first", {"one":1}, "second", 3, null]` .

- Les tableaux JSON utilisent un index de base zéro ; le premier élément d'un tableau se trouve à la position 0. Dans une paire clé:valeur JSON, la clé est une chaîne entre guillemets doubles.
- Une valeur JSON peut être l'une des suivantes :
  - Objet JSON
  - array
  - chaîne
    - Représenté par des guillemets
  - nombre
    - Inclut les nombres entiers, les décimaux et les nombres flottants
  - un booléen
  - null
- Les objets vides et les tableaux vides sont des valeurs JSON valides.
- Les champs JSON sont sensibles à la casse.
- Les espace vides entre les éléments structurels JSON (tel que `{ }, [ ]`) sont ignorés.

Les fonctions JSON Amazon Redshift et la commande COPY Amazon Redshift utilisent les mêmes méthodes pour utiliser des données au format JSON. Pour plus d'informations sur l'utilisation de JSON, consultez Exécution de la commande COPY à partir du format JSON.

Fonction JSON_PARSE

La fonction JSON_PARSE analyse les données au format JSON et les convertit en représentation SUPER.

Pour ingérer dans le type de données SUPER à l'aide de la commande INSERT ou UPDATE, utilisez la fonction JSON_PARSE. Lorsque vous utilisez JSON_PARSE() pour analyser des chaînes JSON en valeurs SUPER, certaines restrictions s'appliquent. Pour plus d'informations, consultez Options d'analyse pour Super.

Syntaxe

```
JSON_PARSE( {json_string | binary_value} )
```

Arguments

json_string

Expression qui renvoie la chaîne JSON sérialisée sous forme de type VARBYTE ou VARCHAR.

binary_value

Valeur binaire du type VARBYTE.

Type de retour

SUPER

Exemples

Pour convertir le tableau JSON `[10001,10002,"abc"]` dans le type de données SUPER, utilisez l'exemple suivant.

```
SELECT JSON_PARSE('[10001,10002,"abc"]');

+---------------------+
|     json_parse      |
+---------------------+
| [10001,10002,"abc"] |
+---------------------+
```

Pour vous assurer que la fonction a converti le tableau JSON dans le type de données SUPER, utilisez l'exemple suivant. Pour de plus amples informations, consultez Fonction JSON_TYPEOF.

```
SELECT JSON_TYPEOF(JSON_PARSE('[10001,10002,"abc"]'));

+-------------+
| json_typeof |
+-------------+
| array       |
+-------------+
```

Fonction CAN_JSON_PARSE

La fonction CAN_JSON_PARSE analyse les données au format JSON et renvoie `true` si le résultat peut être converti en valeur SUPER à l'aide de la fonction JSON_PARSE.

Syntaxe

```
CAN_JSON_PARSE( {json_string | binary_value} )
```

Arguments

json_string

Expression qui renvoie du JSON sérialisé VARCHAR sous forme de formulaire.

binary_value

Valeur binaire du type VARBYTE.

Type de retour

BOOLEAN

Notes d’utilisation

- CAN_JSON_PARSE renvoie false pour les chaînes vides. Elle renvoie NULL lorsque l'argument d'entrée est nul.

Exemples

L'exemple suivant montre que CAN_JSON_PARSE s'exécute sur un tableau JSON correctement formé à l'aide d'une condition CASE. Elle renvoie true. Amazon Redshift exécute donc la fonction JSON_PARSE sur la valeur d'exemple.

```
SELECT CASE
            WHEN CAN_JSON_PARSE('[10001,10002,"abc"]')
            THEN JSON_PARSE('[10001,10002,"abc"]')
        END;

 case
---------------------
'[10001,10002,"abc"]'
```

L'exemple suivant montre l'exécution de CAN_JSON_PARSE sur une valeur autre que le format JSON à l'aide d'une condition CASE. Elle renvoie false. Amazon Redshift renvoie donc le segment dans la clause ELSE de la condition CASE à la place.

```
SELECT CASE
            WHEN CAN_JSON_PARSE('This is a string.')
            THEN JSON_PARSE('This is a string.')
            ELSE 'This is not JSON.'
        END;

 case
--------------------
"This is not JSON."
```

Fonction JSON_SERIALIZE

La fonction JSON_SERIALIZE sérialise une expression SUPER en représentation JSON textuelle pour suivre la norme RFC 8259. Pour plus d'informations sur cette RFC, consultez le format d'échange de données JSON (JavaScript Object Notation).

La limite de taille des données SUPER est approximativement la même que la limite de bloc, et la limite des données VARCHAR est inférieure à la limite de taille des données SUPER. Par conséquent, la fonction JSON_SERIALIZE renvoie une erreur lorsque le format JSON dépasse la limite VARCHAR du système. Pour vérifier la taille d'une expression SUPER, consultez la fonction JSON_SIZE.

Syntaxe

```
JSON_SERIALIZE(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

VARCHAR

> **Note**
>
> La valeur VARCHAR renvoyée est toujours une chaîne JSON non nulle. Si super_expression est NULL, JSON_SERIALIZE renvoie la chaîne JSON. `'null'`

Exemples

Pour sérialiser une valeur SUPER en chaîne, utilisez l'exemple suivant.

```
SELECT JSON_SERIALIZE(JSON_PARSE('[10001,10002,"abc"]'));

+---------------------+
|   json_serialize    |
+---------------------+
| [10001,10002,"abc"] |
+---------------------+
```

Fonction JSON_SERIALIZE_TO_VARBYTE

La fonction JSON_SERIALIZE_TO_VARBYTE convertit une valeur SUPER en chaîne JSON similaire à JSON_SERIALIZE(), mais stockée dans une valeur VARBYTE.

Syntaxe

```
JSON_SERIALIZE_TO_VARBYTE(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

VARBYTE

Exemples

Pour sérialiser une valeur SUPER et renvoyer le résultat au format VARBYTE, utilisez l'exemple suivant.

```
SELECT JSON_SERIALIZE_TO_VARBYTE(JSON_PARSE('[10001,10002,"abc"]'));

+----------------------------------------+
|       json_serialize_to_varbyte        |
+----------------------------------------+
| 5b31303030312c31303030322c22616263225d |
```

```
+----------------------------------------+
```

Pour sérialiser une valeur SUPER et convertir le résultat au format VARCHAR, utilisez l'exemple suivant. Pour de plus amples informations, veuillez consulter Fonction CAST.

```
SELECT CAST((JSON_SERIALIZE_TO_VARBYTE(JSON_PARSE('[10001,10002,"abc"]'))) AS VARCHAR);

+---------------------------+
| json_serialize_to_varbyte |
+---------------------------+
| [10001,10002,"abc"]       |
+---------------------------+
```

Fonctions JSON basées sur du texte

Les fonctions suivantes analysent les valeurs JSON en tant que VARCHAR. Pour analyser le JSON, nous vous recommandons plutôt d'utiliser les fonctions suivantes, qui analysent les valeurs JSON en tant que SUPER. Amazon Redshift analyse les valeurs SUPER plus efficacement que VARCHAR.

- Fonction JSON_PARSE
- Fonction CAN_JSON_PARSE
- Fonction JSON_SERIALIZE
- Fonction JSON_SERIALIZE_TO_VARBYTE

Rubriques

- Fonction IS_VALID_JSON
- Fonction IS_VALID_JSON_ARRAY
- Fonction JSON_ARRAY_LENGTH
- Fonction JSON_EXTRACT_ARRAY_ELEMENT_TEXT
- Fonction JSON_EXTRACT_PATH_TEXT

Fonction IS_VALID_JSON

> **Note**
>
> CAN_JSON_PARSE et ses fonctions associées analysent les valeurs JSON en tant que SUPER, qu'Amazon Redshift analyse plus efficacement que VARCHAR.

> Au lieu d'utiliser IS_VALID_JSON, nous vous recommandons de valider vos chaînes JSON à l'aide du. Fonction CAN_JSON_PARSE

La fonction IS_VALID_JSON valide une chaîne JSON. La fonction renvoie la valeur booléenne `true` si la chaîne est une chaîne JSON correctement formée ou `false` dans le cas contraire. Pour valider un tableau JSON, utilisez Fonction IS_VALID_JSON_ARRAY

Pour plus d'informations, consultez Fonctions JSON.

Syntaxe

```
IS_VALID_JSON('json_string')
```

Arguments

json_string

Chaîne ou expression ayant pour valeur une chaîne JSON.

Type de retour

BOOLEAN

Exemples

Pour créer une table et insérer des chaînes JSON à des fins de test, utilisez l'exemple suivant.

```
CREATE TABLE test_json(id int IDENTITY(0,1), json_strings VARCHAR);

-- Insert valid JSON strings --
INSERT INTO test_json(json_strings) VALUES
('{"a":2}'),
('{"a":{"b":{"c":1}}}'),
('{"a": [1,2,"b"]}');

-- Insert invalid JSON strings --
INSERT INTO test_json(json_strings) VALUES
('{{}}'),
('{1:"a"}'),
```

```
('[1,2,3]');
```

Pour valider les chaînes dans l'exemple précédent, utilisez l'exemple suivant.

```
SELECT id, json_strings, IS_VALID_JSON(json_strings)
FROM test_json
ORDER BY id;

+----+---------------------+---------------+
| id |    json_strings     | is_valid_json |
+----+---------------------+---------------+
|  0 | {"a":2}             | true          |
|  4 | {"a":{"b":{"c":1}}} | true          |
|  8 | {"a": [1,2,"b"]}    | true          |
| 12 | {{}}                | false         |
| 16 | {1:"a"}             | false         |
| 20 | [1,2,3]             | false         |
+----+---------------------+---------------+
```

Fonction IS_VALID_JSON_ARRAY

> **Note**
>
> JSON_PARSE et ses fonctions associées analysent les valeurs JSON en tant que SUPER, qu'Amazon Redshift analyse plus efficacement que VARCHAR.
> Au lieu d'utiliser IS_VALID_JSON_ARRAY, nous vous recommandons d'analyser vos chaînes JSON en utilisant le pour obtenir une valeur SUPER. Fonction JSON_PARSE Utilisez ensuite la Fonction IS_ARRAY fonction pour confirmer que le tableau est correctement formé.

La fonction IS_VALID_JSON_ARRAY valide un tableau JSON. La fonction renvoie la valeur booléenne `true` si le tableau est un tableau JSON correctement formé ou `false` dans le cas contraire. Pour valider un tableau JSON, utilisez Fonction IS_VALID_JSON

Pour plus d'informations, consultez Fonctions JSON.

Syntaxe

```
IS_VALID_JSON_ARRAY('json_array')
```

Arguments

json_array

Chaîne ou expression ayant pour valeur un tableau JSON.

Type de retour

BOOLEAN

Exemples

Pour créer une table et insérer des chaînes JSON à des fins de test, utilisez l'exemple suivant.

```
CREATE TABLE test_json_arrays(id int IDENTITY(0,1), json_arrays VARCHAR);

-- Insert valid JSON array strings --
INSERT INTO test_json_arrays(json_arrays)
VALUES('[]'),
('["a","b"]'),
('["a",["b",1,["c",2,3,null]]]');

-- Insert invalid JSON array strings --
INSERT INTO test_json_arrays(json_arrays)
VALUES('{"a":1}'),
('a'),
('[1,2,]');
```

Pour valider les chaînes dans l'exemple précédent, utilisez l'exemple suivant.

```
SELECT json_arrays, IS_VALID_JSON_ARRAY(json_arrays)
FROM test_json_arrays ORDER BY id;

+------------------------------+---------------------+
|         json_arrays          | is_valid_json_array |
+------------------------------+---------------------+
| []                           | true                |
| ["a","b"]                    | true                |
| ["a",["b",1,["c",2,3,null]]] | true                |
| {"a":1}                      | false               |
| a                            | false               |
| [1,2,]                       | false               |
```

```
+-----------------------------+--------------------+
```

Fonction JSON_ARRAY_LENGTH

> **Note**
>
> JSON_PARSE et ses fonctions associées analysent les valeurs JSON en tant que SUPER, qu'Amazon Redshift analyse plus efficacement que VARCHAR.
> Au lieu d'utiliser JSON_ARRAY_LENGTH, nous vous recommandons d'analyser vos chaînes JSON en utilisant le pour obtenir une valeur SUPER. Fonction JSON_PARSE Ensuite, utilisez le Fonction get_array_length pour obtenir la longueur de votre tableau.

La fonction JSON_ARRAY_LENGTH renvoie le nombre d'éléments du tableau externe d'une chaîne JSON. Si l'argument null_if_invalid a la valeur `true` et que la chaîne JSON n'est pas valide, la fonction renvoie `NULL` au lieu de renvoyer une erreur.

Pour de plus amples informations, veuillez consulter Fonctions JSON.

Syntaxe

```
JSON_ARRAY_LENGTH('json_array' [, null_if_invalid ] )
```

Arguments

json_array

Tableau JSON au bon format.

null_if_invalid

(Facultatif) Valeur `BOOLEAN` qui spécifie s'il faut renvoyer `NULL` quand la chaîne JSON en entrée n'est pas valide au lieu de renvoyer une erreur. Pour renvoyer `NULL` si la chaîne JSON n'est pas valide, spécifiez `true` (`t`). Pour renvoyer une erreur si la chaîne JSON n'est pas valide, spécifiez `false` (`f`). L'argument par défaut est `false`.

Type de retour

`INTEGER`

Exemples

Pour renvoyer le nombre d'éléments du tableau, utilisez l'exemple suivant.

```
SELECT JSON_ARRAY_LENGTH('[11,12,13,{"f1":21,"f2":[25,26]},14]');

+-------------------+
| json_array_length |
+-------------------+
|                 5 |
+-------------------+
```

Pour renvoyer une erreur si la chaîne JSON n'est pas valide, utilisez l'exemple suivant.

```
SELECT JSON_ARRAY_LENGTH('[11,12,13,{"f1":21,"f2":[25,26]},14');

ERROR: invalid json array object [11,12,13,{"f1":21,"f2":[25,26]},14
```

Pour définir null_if_invalid sur true afin que l'instruction renvoie NULL au lieu de renvoyer une erreur en cas de chaîne JSON non valide, utilisez l'exemple suivant.

```
SELECT JSON_ARRAY_LENGTH('[11,12,13,{"f1":21,"f2":[25,26]},14',true);

+-------------------+
| json_array_length |
+-------------------+
| NULL              |
+-------------------+
```

Fonction JSON_EXTRACT_ARRAY_ELEMENT_TEXT

> **Note**
>
> JSON_PARSE et ses fonctions associées analysent les valeurs JSON en tant que SUPER, qu'Amazon Redshift analyse plus efficacement que VARCHAR.
> Au lieu d'utiliser JSON_EXTRACT_ARRAY_ELEMENT_TEXT, nous vous recommandons d'analyser vos chaînes JSON en utilisant le pour obtenir une valeur SUPER. Fonction JSON_PARSE Ensuite, interrogez l'élément souhaité à l'aide de son index de tableau, en utilisant la `value[element position]` syntaxe. Pour plus d'informations sur l'interrogation

des éléments d'un tableau dans des valeurs SUPER, consultez. Interrogation de données semi-structurées

La fonction JSON_EXTRACT_ARRAY_ELEMENT_TEXT renvoie un élément de tableau JSON dans le tableau le plus externe d'une chaîne JSON, à l'aide d'un index de base zéro. Le premier élément d'un tableau est à la position 0. Si l'indice est négatif ou hors limites, JSON_EXTRACT_ARRAY_ELEMENT_TEXT renvoie. NULL Si l'argument null_if_invalid a la valeur TRUE et que la chaîne JSON n'est pas valide, la fonction renvoie NULL au lieu de renvoyer une erreur.

Pour de plus amples informations, veuillez consulter Fonctions JSON.

Syntaxe

```
JSON_EXTRACT_ARRAY_ELEMENT_TEXT('json string', pos [, null_if_invalid ] )
```

Arguments

json_string

Chaîne JSON au bon format.

pos

INTEGER représentant l'index de l'élément de tableau à renvoyer, à l'aide d'un index de tableau de base zéro.

null_if_invalid

(Facultatif) Valeur BOOLEAN qui spécifie s'il faut renvoyer NULL quand la chaîne JSON en entrée n'est pas valide au lieu de renvoyer une erreur. Pour renvoyer NULL si la chaîne JSON n'est pas valide, spécifiez `true` (`t`). Pour renvoyer une erreur si la chaîne JSON n'est pas valide, spécifiez `false` (`f`). L'argument par défaut est `false`.

Type de retour

VARCHAR

Chaîne VARCHAR représentant l'élément de tableau JSON référencé par pos.

Exemples

Pour renvoyer l'élément de tableau à la position 2, qui est le troisième élément d'un index de tableau de base zéro, utilisez l'exemple suivant.

```
SELECT JSON_EXTRACT_ARRAY_ELEMENT_TEXT('[111,112,113]', 2);

+--------------------------------+
| json_extract_array_element_text |
+--------------------------------+
|                            113 |
+--------------------------------+
```

Pour renvoyer une erreur si la chaîne JSON n'est pas valide, utilisez l'exemple suivant.

```
SELECT JSON_EXTRACT_ARRAY_ELEMENT_TEXT('["a",["b",1,["c",2,3,null,]]]',1);

ERROR: invalid json array object ["a",["b",1,["c",2,3,null,]]]
```

Pour définir null_if_invalid sur true afin que l'instruction renvoie NULL au lieu de renvoyer une erreur en cas de chaîne JSON non valide, utilisez l'exemple suivant.

```
SELECT JSON_EXTRACT_ARRAY_ELEMENT_TEXT('["a",["b",1,["c",2,3,null,]]]',1,true);

+--------------------------------+
| json_extract_array_element_text |
+--------------------------------+
| NULL                           |
+--------------------------------+
```

Examinez les exemples d'énoncés suivants. Si la chaîne JSON ou l'index fourni est NULL, JSON_EXTRACT_ARRAY_ELEMENT_TEXT renvoie NULL quelle que soit la valeur des autres paramètres.

```
--Statement where json_string is NULL.
SELECT json_extract_array_element_text(NULL, 0)

 json_extract_array_element_text
--------------------------------
                            NULL
```

```
--Statement where pos is NULL and json_string is invalid JSON.
SELECT json_extract_array_element_text('invalid_json', NULL);

 json_extract_array_element_text
---------------------------------
                            NULL

--Statement where json_string is NULL and null_if_invalid is FALSE.
SELECT json_extract_array_element_text(NULL, 0, FALSE);

 json_extract_array_element_text
---------------------------------
                            NULL
```

Examinez les exemples d'énoncés suivants. Lorsque null_if_invalid est TRUE, JSON_EXTRACT_ARRAY_ELEMENT_TEXT renvoie NULL lorsque json_string est un JSON non valide. Si null_if_invalid vaut FALSE ou n'est pas défini, la fonction renvoie une erreur lorsque json_string n'est pas valide.

```
--Statement with invalid JSON where null_if_invalid is TRUE.
SELECT json_extract_array_element_text('invalid_json', 0, TRUE);

 json_extract_array_element_text
---------------------------------
                            NULL

--Statement with invalid JSON where null_if_invalid is FALSE.
SELECT json_extract_array_element_text('invalid_json', 0);

ERROR:  JSON parsing error
```

Prenons l'exemple suivant, où json_string est un JSON valide et pos fait référence à une valeur JSON. `null` Dans ce cas, JSON_EXTRACT_ARRAY_ELEMENT_TEXT renvoie NULL, quelle que soit la valeur de null_if_invalid.

```
--Statement selecting a null value.
SELECT json_extract_array_element_text('[null]', 0);

  json_extract_array_element_text
---------------------------------
                             NULL
```

Fonction JSON_EXTRACT_PATH_TEXT

> **Note**
>
> JSON_PARSE et ses fonctions associées analysent les valeurs JSON en tant que SUPER, qu'Amazon Redshift analyse plus efficacement que VARCHAR.
> Au lieu d'utiliser JSON_EXTRACT_PATH_TEXT, nous vous recommandons d'analyser vos chaînes JSON en utilisant le pour obtenir une valeur SUPER. Fonction JSON_PARSE
> Ensuite, recherchez l'élément souhaité à l'aide de la `value.attribute` syntaxe. Pour plus d'informations sur l'interrogation des éléments d'un tableau dans des valeurs SUPER, consultez. Interrogation de données semi-structurées

La fonction JSON_EXTRACT_PATH_TEXT renvoie la valeur de la paire clé-valeur référencée par une série d'éléments de chemin dans une chaîne JSON. Le chemin d'accès JSON peut s'imbriquer à une profondeur de près de cinq niveaux. Les éléments de chemin d'accès sont sensible à la casse. Si un élément de chemin n'existe pas dans la chaîne JSON, JSON_EXTRACT_PATH_TEXT renvoie `NULL`.

Si l'argument null_if_invalid a la valeur `TRUE` et que la chaîne JSON n'est pas valide, la fonction renvoie `NULL` au lieu de renvoyer une erreur.

JSON_EXTRACT_PATH_TEXT a une taille de données maximale de 64 Ko. Ainsi, si un enregistrement JSON est supérieur à 64 Ko, son traitement avec JSON_EXTRACT_PATH_TEXT entraîne une erreur.

Pour plus d'informations sur les fonctions JSON supplémentaires, consultez Fonctions JSON. Pour plus d'informations sur l'utilisation de JSON, consultez Exécution de la commande COPY à partir du format JSON.

Syntaxe

```
JSON_EXTRACT_PATH_TEXT('json_string', 'path_elem' [,'path_elem'[, …] ]
 [, null_if_invalid ] )
```

Arguments

json_string

Chaîne JSON au bon format.

path_elem

Élément de chemin d'accès dans une chaîne JSON. Un élément de chemin d'accès est obligatoire. Des éléments de chemin supplémentaires peuvent être spécifiés, jusqu'à une profondeur de cinq niveaux.

null_if_invalid

(Facultatif) Valeur BOOLEAN qui spécifie s'il faut renvoyer NULL quand la chaîne JSON en entrée n'est pas valide au lieu de renvoyer une erreur. Pour renvoyer NULL si la chaîne JSON n'est pas valide, spécifiez TRUE (t). Pour renvoyer une erreur si la chaîne JSON n'est pas valide, spécifiez FALSE (f). L'argument par défaut est FALSE.

Dans une chaîne JSON, Amazon Redshift reconnaît \n comme un caractère de nouvelle ligne et \t comme un caractère de tabulation. Pour charger une barre oblique inverse, précédez-la d'une barre oblique inverse (\\). Pour plus d'informations, consultez Caractères d'échappement dans JSON.

Type de retour

VARCHAR

Chaîne VARCHAR représentant la valeur JSON référencée par les éléments de chemin.

Exemples

Pour renvoyer la valeur du chemin 'f4', 'f6', utilisez l'exemple suivant.

```
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}','f4', 'f6');

+------------------------+
| json_extract_path_text |
+------------------------+
| star                   |
+------------------------+
```

Pour renvoyer une erreur si la chaîne JSON n'est pas valide, utilisez l'exemple suivant.

```
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}','f4', 'f6');
```

```
ERROR: invalid json object {"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}
```

Pour attribuer à null_if_invalid la valeur TRUE, afin que l'instruction renvoie un JSON non valide au lieu de renvoyer une erreur, utilisez l'NULLexemple suivant.

```
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}','f4',
 'f6',true);

+------------------------+
| json_extract_path_text |
+------------------------+
| NULL                   |
+------------------------+
```

Prenons l'exemple suivant, qui sélectionne la valeur du chemin`'farm', 'barn', 'color'`, où la valeur récupérée est au troisième niveau, utilisez l'exemple suivant. Cet exemple est formaté avec un outil de validation JSON, pour le rendre plus facile à lire.

```
SELECT JSON_EXTRACT_PATH_TEXT('{
    "farm": {
        "barn": {
            "color": "red",
            "feed stocked": true
        }
    }
}', 'farm', 'barn', 'color');
+------------------------+
| json_extract_path_text |
+------------------------+
| red                    |
+------------------------+
```

Pour renvoyer NULL si l'élément `'color'` est manquant, utilisez l'exemple suivant. Cet exemple est formaté avec un outil de validation JSON.

```
SELECT JSON_EXTRACT_PATH_TEXT('{
    "farm": {
        "barn": {}
    }
}', 'farm', 'barn', 'color');

+------------------------+
```

```
| json_extract_path_text |
+------------------------+
| NULL                   |
+------------------------+
```

Si la chaîne JSON est valide, la tentative d'extraction d'un élément manquant renvoie NULL.

Pour renvoyer la valeur du chemin `'house', 'appliances', 'washing machine', 'brand'`, utilisez l'exemple suivant.

```
SELECT JSON_EXTRACT_PATH_TEXT('{
  "house": {
    "address": {
      "street": "123 Any St.",
      "city": "Any Town",
      "state": "FL",
      "zip": "32830"
    },
    "bathroom": {
      "color": "green",
      "shower": true
    },
    "appliances": {
      "washing machine": {
        "brand": "Any Brand",
        "color": "beige"
      },
      "dryer": {
        "brand": "Any Brand",
        "color": "white"
      }
    }
  }
}', 'house', 'appliances', 'washing machine', 'brand');

+------------------------+
| json_extract_path_text |
+------------------------+
| Any Brand              |
+------------------------+
```

L'exemple suivant crée un exemple de table et le remplit avec des valeurs SUPER, puis renvoie la valeur du chemin `'f2'` pour les deux lignes.

```
CREATE TABLE json_example(id INT, json_text SUPER);

INSERT INTO json_example VALUES
(1, JSON_PARSE('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}')),
(2, JSON_PARSE('{
    "farm": {
        "barn": {
            "color": "red",
            "feed stocked": true
        }
    }
}'));

SELECT * FROM json_example;
id          | json_text
------------+-------------------------------------------
1           | {"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}
2           | {"farm":{"barn":{"color":"red","feed stocked":true}}}


SELECT id, JSON_EXTRACT_PATH_TEXT(JSON_SERIALIZE(json_text), 'f2') FROM json_example;

id          | json_text
------------+-------------------------------------------
1           | {"f3":1}
2           |
```

Examinez les exemples d'énoncés suivants. Le path_elem fourni est NULL, donc JSON_EXTRACT_PATH_TEXT renvoie NULL quelle que soit la valeur des autres paramètres.

```
--Statement where path_elem is NULL and json_string is valid JSON.
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}',NULL);

 json_extract_path_text
------------------------
                  NULL

--Statement where only one path_elem is NULL.
SELECT JSON_EXTRACT_PATH_TEXT('{"f2":{"f3":1},"f4":{"f5":99,"f6":"star"}}','f4',NULL);

 json_extract_path_text
------------------------
                  NULL
```

```
--Statement where path_elem is NULL and json_string is invalid JSON.
SELECT json_extract_path_text('invalid_json', NULL);

 json_extract_path_text
------------------------
                  NULL

--Statement where path_elem is NULL and null_if_invalid is FALSE.
SELECT json_extract_path_text(NULL, 0, FALSE);

 json_extract_path_text
------------------------
                  NULL
```

Examinez les exemples d'énoncés suivants. Lorsque null_if_invalid est TRUE, JSON_EXTRACT_PATH_TEXT renvoie NULL lorsque json_string est un JSON non valide. Si null_if_invalid vaut FALSE ou n'est pas défini, la fonction renvoie une erreur lorsque json_string n'est pas valide.

```
--Statement with invalid JSON where null_if_invalid is TRUE.
SELECT json_extract_path_text('invalid_json', 0, TRUE);

 json_extract_path_text
------------------------
                  NULL

--Statement with invalid JSON where null_if_invalid is FALSE.
SELECT json_extract_path_text('invalid_json', 0, FALSE);

ERROR:  JSON parsing error
```

Prenons les exemples suivants, où json_string est un JSON valide et path_elem fait référence à une valeur JSON. `null` Dans ce cas, JSON_EXTRACT_PATH_TEXT renvoie NULL. De même, lorsque path_elem fait référence à une valeur inexistante, JSON_EXTRACT_PATH_TEXT renvoie NULL, quelle que soit la valeur de null_if_invalid.

```
--Statement selecting a null value.
SELECT json_extract_path_text('[null]', 0);

  json_extract_path_text
-------------------------
```

```
                    NULL

--Statement selecting a non-existing value.
SELECT json_extract_path_text('{}', 'a');

  json_extract_path_text
-------------------------
                    NULL
```

Solutions de machine learning

En utilisant le machine learning (ML) d'Amazon Redshift, vous pouvez entraîner des modèles de machine learning en utilisant des instructions SQL et les invoquer dans des requêtes SQL pour la prédiction. L'explicabilité du modèle Amazon Redshift inclut des valeurs d'importance des fonctions pour vous aider à comprendre comment chaque attribut de vos données d'entraînement contribue au résultat prédit.

Vous trouverez ci-dessous les descriptions des fonctions de machine learning pour SQL prises en charge par Amazon Redshift.

Rubriques

- Fonction EXPLAIN_MODEL

Fonction EXPLAIN_MODEL

La fonction EXPLAIN_MODEL renvoie un type de données SUPER qui contient un rapport d'explicabilité du modèle au format JSON. Le rapport d'explicabilité contient des informations sur la valeur Shapley pour toutes les fonctions du modèle.

La fonction EXPLAIN_MODEL ne prend actuellement en charge que les modèles AUTO ON ou AUTO OFF. XGBoost

Lorsque le rapport d'explicabilité n'est pas disponible, la fonction renvoie les statuts affichant la progression du modèle. Cela inclut `Waiting for training job to complete`, `Waiting for processing job to complete` et `Processing job failed`.

Lorsque vous exécutez l'instruction CREATE MODEL, l'état de l'explication devient `Waiting for training job to complete`. Lorsque le modèle a été formé et qu'une demande d'explication est envoyée, l'état de l'explication devient `Waiting for processing job to complete`. Lorsque

l'explication du modèle est terminée avec succès, le rapport d'explicabilité complet est disponible. Sinon, l'état devient `Processing job failed`.

Lorsque vous exécutez l'instruction CREATE MODEL, vous pouvez utiliser le paramètre facultatif `MAX_RUNTIME` afin de spécifier la durée maximale de l'entraînement. Une fois ce délai de création de modèle atteint, Amazon Redshift arrête la création du modèle. Si vous atteignez ce délai pendant que vous créez un modèle Autopilot, Amazon Redshift renvoie le meilleur modèle obtenu jusque-là. Sachant que l'explicabilité du modèle devient disponible une fois l'entraînement du modèle terminé, si la valeur de durée définie pour `MAX_RUNTIME` est basse, il est possible que le rapport d'explicabilité ne soit pas disponible. La durée d'entraînement varie en fonction de la complexité du modèle, de la taille des données et d'autres facteurs.

Syntaxe

```
EXPLAIN_MODEL ('schema_name.model_name')
```

Argument

nom_schéma

Nom du schéma. Si aucun nom de schéma n'est spécifié, le schéma actuel est sélectionné.

model_name

Nom du modèle. Le nom du modèle d'un schéma doit être unique.

Type de retour

La fonction EXPLAIN_MODEL renvoie un type de données SUPER, comme indiqué ci-dessous.

```
{"version":"1.0","explanations":{"kernel_shap":{"label0":{"global_shap_values":
{"x0":0.05,"x1":0.10,"x2":0.30,"x3":0.15},"expected_value":0.50}}}}
```

Exemples

L'exemple suivant renvoie l'état de l'explication `waiting for training job to complete`.

```
select explain_model('customer_churn_auto_model');
                explain_model
-------------------------------------------------------
```

```
{"explanations":"waiting for training job to complete"}
(1 row)
```

Lorsque l'explication du modèle est terminée avec succès, le rapport d'explicabilité complet est disponible comme suit.

```
select explain_model('customer_churn_auto_model');
                                        explain_model
-----------------------------------------------------------------------------------------------
{"version":"1.0","explanations":{"kernel_shap":{"label0":{"global_shap_values":
{"x0":0.05386043365892927,"x1":0.10801289723274592,"x2":0.23227865827017378,"x3":0.067668513394
(1 row)
```

Étant donné que la fonction EXPLAIN_MODEL renvoie le type de données SUPER, vous pouvez interroger le rapport d'explicabilité. En faisant cela, vous pouvez extraire des valeurs `global_shap_values`, `expected_value` ou des valeurs Shapley spécifiques aux fonctions.

L'exemple suivant montre comment extraire `global_shap_values` pour le modèle.

```
select json_table.report.explanations.kernel_shap.label0.global_shap_values from
 (select explain_model('customer_churn_auto_model') as report) as json_table;
                                                    global_shap_values
-----------------------------------------------------------------------------------------------
{"state":0.10983770427197151,"account_length":0.1772441398408543,"area_code":0.0862682396863959
(1 row)
```

L'exemple suivant montre comment extraire `global_shap_values` pour la fonction x0.

```
select json_table.report.explanations.kernel_shap.label0.global_shap_values.x0 from
 (select explain_model('customer_churn_auto_model') as report) as json_table;
         x0
------------------------
  0.05386043365892927
(1 row)
```

Si le modèle est créé dans un schéma spécifique et que vous avez accès au modèle créé, vous pouvez interroger l'explication du modèle comme indiqué ci-dessous.

```
-- Check the current schema
SHOW search_path;
```

```
   search_path
-----------------
  $user, public
(1 row)
-- If you have the privilege to access the model explanation
-- in `test_schema`
SELECT explain_model('test_schema.test_model_name');
                      explain_model
-------------------------------------------------------------
{"explanations":"waiting for training job to complete"}
(1 row)
```

Fonctions mathématiques

Rubriques

- Symboles d'opérateurs mathématiques
- Fonction ABS
- Fonction ACOS
- Fonction ASIN
- Fonction ATAN
- ATAN2 fonction
- Fonction CBRT
- Fonction CEILING (ou CEIL)
- Fonction COS
- Fonction COT
- Fonction DEGREES
- Fonction DEXP
- DLOG1 fonction
- DLOG10 fonction
- Fonction EXP
- Fonction FLOOR
- Fonction LN
- Fonction LOG
- Fonction MOD

- Fonction PI
- Fonction POWER
- Fonction RADIANS
- Fonction RANDOM
- Fonction ROUND
- Fonction SIN
- Fonction SIGN
- Fonction SQRT
- Fonction TAN
- Fonction TRUNC

Cette section décrit les fonctions et opérateurs mathématiques pris en charge par Amazon Redshift.

Symboles d'opérateurs mathématiques

Le tableau suivant répertorie les opérateurs mathématiques pris en charge.

Opérateurs pris en charge

Opérateur	Description	Exemple	Résultat
+	addition	2 + 3	5
-	soustraction	2-3	-1
*	multiplication	2 * 3	6
/	division	4 / 2	2
%	modulo	5 % 4	1
^	puissance	2.0 ^ 3.0	8
|/	racine carrée	| / 25.0	5

Opérateur	Description	Exemple	Résultat
||/	racine cubique	|| / 27.0	3
@	valeur absolue	@ -5.0	5
<<	au niveau du bit de décalage gauche	1 << 4	16
>>	au niveau du bit de décalage droit	8 >> 2	2
&	au niveau du bit et	8 & 2	0

Exemples

Les exemples suivants utilisent l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour calculer la commission payée plus 2,00 USD de traitement pour une transaction donnée, utilisez l'exemple suivant.

```
SELECT
    commission,
    (commission + 2.00) AS comm
FROM
    sales
WHERE
    salesid = 10000;

+------------+-------+
| commission | comm  |
+------------+-------+
```

```
|      28.05 | 30.05 |
+------------+-------+
```

Pour calculer 20 % du prix de vente pour une transaction donnée, utilisez l'exemple suivant.

```
SELECT pricepaid, (pricepaid * .20) as twentypct
FROM sales
WHERE salesid=10000;

+-----------+-----------+
| pricepaid | twentypct |
+-----------+-----------+
|       187 |      37.4 |
+-----------+-----------+
```

Pour prévoir les ventes de billets selon un modèle de croissance continue, utilisez l'exemple suivant. Dans cet exemple, la sous-requête renvoie le nombre de billets vendus en 2008. Ce résultat est multiplié de manière exponentielle par un taux de croissance continue de 5 % sur 10 ans.

```
SELECT (SELECT SUM(qtysold) FROM sales, date
WHERE sales.dateid=date.dateid AND year=2008)^((5::float/100)*10) AS qty10years;

+------------------+
|    qty10years    |
+------------------+
| 587.664019657491 |
+------------------+
```

Pour rechercher le prix total payé et la commission sur les ventes avec un ID de date supérieur ou égal à 2 000, utilisez l'exemple suivant. Puis soustrayez la commission totale du prix total payé.

```
SELECT SUM(pricepaid) AS sum_price, dateid,
SUM(commission) AS sum_comm, (SUM(pricepaid) - SUM(commission)) AS value
FROM sales
WHERE dateid >= 2000
GROUP BY dateid
ORDER BY dateid
LIMIT 10;

+-----------+--------+----------+-----------+
| sum_price | dateid | sum_comm |   value   |
+-----------+--------+----------+-----------+
```

```
|    305885 |   2000 | 45882.75 | 260002.25 |
|    316037 |   2001 | 47405.55 | 268631.45 |
|    358571 |   2002 | 53785.65 | 304785.35 |
|    366033 |   2003 | 54904.95 | 311128.05 |
|    307592 |   2004 |  46138.8 |  261453.2 |
|    333484 |   2005 |  50022.6 |  283461.4 |
|    317670 |   2006 |  47650.5 |  270019.5 |
|    351031 |   2007 | 52654.65 | 298376.35 |
|    313359 |   2008 | 47003.85 | 266355.15 |
|    323675 |   2009 | 48551.25 | 275123.75 |
+-----------+--------+----------+-----------+
```

Fonction ABS

ABS calcule la valeur absolue d'un nombre, où ce nombre peut être littéral ou une expression qui a pour valeur un nombre.

Syntaxe

```
ABS(number)
```

Arguments

number

Nombre ou expression ayant pour valeur un nombre. Il peut être de type SMALLINT, INTEGER, BIGINT, DECIMAL, FLOAT4, FLOAT8 ou SUPER.

Type de retour

ABS renvoie le même type de données que sont argument.

Exemples

Pour calculer la valeur absolue de -38, utilisez l'exemple suivant.

```
SELECT ABS(-38);

+-----+
| abs |
+-----+
|  38 |
```

```
+-----+
```

Pour calculer la valeur absolue de `(14-76)`, utilisez l'exemple suivant.

```
SELECT ABS(14-76);

+-----+
| abs |
+-----+
|  62 |
+-----+
```

Fonction ACOS

ACOS est une fonction trigonométrique qui renvoie l'arc cosinus d'un nombre. La valeur de retour est exprimée en radians et se situe entre `0` et `PI`.

Syntaxe

```
ACOS(number)
```

Arguments

number

Le paramètre d'entrée est un nombre `DOUBLE PRECISION`.

Type de retour

`DOUBLE PRECISION`

Exemples

Pour renvoyer l'arc cosinus de `-1`, utilisez l'exemple suivant.

```
SELECT ACOS(-1);

+-------------------+
|       acos        |
+-------------------+
| 3.141592653589793 |
```

```
+-------------------+
```

Pour convertir l'arc cosinus de `.5` en nombre de degrés équivalent, utilisez l'exemple suivant.

```
SELECT (ACOS(.5) * 180/(SELECT PI())) AS degrees;

+-------------------+
|      degrees      |
+-------------------+
| 60.00000000000001 |
+-------------------+
```

Fonction ASIN

ASIN est une fonction trigonométrique qui renvoie l'arc sinus d'un nombre. La valeur de retour est exprimée en radians et se situe entre `PI/2` et `-PI/2`.

Syntaxe

```
ASIN(number)
```

Arguments

number

Le paramètre d'entrée est un nombre `DOUBLE PRECISION`.

Type de retour

`DOUBLE PRECISION`

Exemples

Pour renvoyer l'arc sinus de 1, utilisez l'exemple suivant.

```
SELECT ASIN(1) AS halfpi;

+--------------------+
|       halfpi       |
+--------------------+
| 1.5707963267948966 |
```

```
+--------------------+
```

Pour convertir l'arc sinus de `.5` en nombre de degrés équivalent, utilisez l'exemple suivant.

```
SELECT (ASIN(.5) * 180/(SELECT PI())) AS degrees;

+--------------------+
|      degrees       |
+--------------------+
| 30.000000000000004 |
+--------------------+
```

Fonction ATAN

ATAN est une fonction trigonométrique qui renvoie l'arc tangente d'un nombre. La valeur de retour est exprimée en radians et se situe entre `-PI` et `PI`.

Syntaxe

```
ATAN(number)
```

Arguments

number

Le paramètre d'entrée est un nombre `DOUBLE PRECISION`.

Type de retour

`DOUBLE PRECISION`

Exemples

Pour renvoyer l'arc tangente de 1 et le multiplier par 4, utilisez l'exemple suivant.

```
SELECT ATAN(1) * 4 AS pi;

+-------------------+
|        pi         |
+-------------------+
| 3.141592653589793 |
```

```
+-------------------+
```

Pour convertir l'arc tangente de 1 en nombre de degrés équivalent, utilisez l'exemple suivant.

```
SELECT (ATAN(1) * 180/(SELECT PI())) AS degrees;

+---------+
| degrees |
+---------+
|      45 |
+---------+
```

ATAN2 fonction

ATAN2 est une fonction trigonométrique qui renvoie l'arc tangente d'un nombre divisé par un autre nombre. La valeur de retour est exprimée en radians et se situe entre `PI/2` et `-PI/2`.

Syntaxe

```
ATAN2(number1, number2)
```

Arguments

number1

Nombre `DOUBLE PRECISION`.

number2

Nombre `DOUBLE PRECISION`.

Type de retour

`DOUBLE PRECISION`

Exemples

Pour renvoyer l'arc tangente de `2/2` et le multiplier par 4, utilisez l'exemple suivant.

```
SELECT ATAN2(2,2) * 4 AS PI;

+-------------------+
```

```
|        pi         |
+-------------------+
| 3.141592653589793 |
+-------------------+
```

Pour convertir l'arc tangente de `1/0` (qui équivaut à 0) en nombre de degrés équivalent, utilisez l'exemple suivant.

```
SELECT (ATAN2(1,0) * 180/(SELECT PI())) AS degrees;

+---------+
| degrees |
+---------+
|      90 |
+---------+
```

Fonction CBRT

La fonction CBRT est une fonction mathématique qui calcule la racine cubique d'un nombre donné.

Syntaxe

```
CBRT(number)
```

Arguments

CBRT accepte un nombre `DOUBLE PRECISION` comme argument.

Type de retour

`DOUBLE PRECISION`

Exemples

L'exemple suivant utilise l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour calculer la racine cubique de la commission payée pour une transaction donnée, utilisez l'exemple suivant.

```
SELECT CBRT(commission) FROM sales WHERE salesid=10000;

+--------------------+
```

```
|        cbrt        |
+--------------------+
| 3.0383953904884344 |
+--------------------+
```

Fonction CEILING (ou CEIL)

La fonction CEILING (ou CEIL) permet d'arrondir un nombre jusqu'au nombre entier supérieur suivant. (Le Fonction FLOOR arrondit un nombre au nombre entier inférieur suivant.)

Syntaxe

```
{CEIL | CEILING}(number)
```

Arguments

number

Nombre ou expression ayant pour valeur un nombre. Il peut être de type SMALLINT, INTEGER, BIGINT, DECIMAL, FLOAT4, FLOAT8 ou SUPER.

Type de retour

CEILING et CEIL renvoient le même type de données que leur argument.

Quand l'entrée est de type SUPER, la sortie conserve le même type dynamique que l'entrée, tandis que le type statique reste le type SUPER. Quand le type dynamique de SUPER n'est pas un nombre, Amazon Redshift renvoie une valeur null.

Exemples

L'exemple suivant utilise l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour calculer le plafond de la commission payée pour une transaction de vente donnée, utilisez l'exemple suivant.

```
SELECT CEILING(commission) FROM sales
WHERE salesid=10000;

+---------+
| ceiling |
```

```
+---------+
|      29 |
+---------+
```

Fonction COS

COS est une fonction trigonométrique qui renvoie le cosinus d'un nombre. La valeur de retour est exprimée en radians et se situe entre `-1` et `1`, inclus.

Syntaxe

```
COS(double_precision)
```

Arguments

number

Le paramètre d'entrée est un nombre `DOUBLE PRECISION`.

Type de retour

La fonction COS renvoie un nombre `DOUBLE PRECISION`.

Exemples

Pour renvoyer le cosinus de `0`, utilisez l'exemple suivant.

```
SELECT COS(0);

+-----+
| cos |
+-----+
|   1 |
+-----+
```

Pour renvoyer le cosinus de `pi`, utilisez l'exemple suivant.

```
SELECT COS(PI());

+-----+
| cos |
+-----+
```

```
|  -1 |
+-----+
```

Fonction COT

COT est une fonction trigonométrique qui renvoie la cotangente d'un nombre. Le paramètre d'entrée doit être différent de zéro.

Syntaxe

```
COT(number)
```

Argument

number

Le paramètre d'entrée est un nombre DOUBLE PRECISION.

Type de retour

DOUBLE PRECISION

Exemples

Pour renvoyer la cotangente de 1, utilisez l'exemple suivant.

```
SELECT COT(1);

+--------------------+
|        cot         |
+--------------------+
| 0.6420926159343306 |
+--------------------+
```

Fonction DEGREES

Convertit un angle en radians en son équivalent en degrés.

Syntaxe

```
DEGREES(number)
```

Argument

number

Le paramètre d'entrée est un nombre DOUBLE PRECISION.

Type de retour

DOUBLE PRECISION

Exemples

Pour renvoyer l'équivalent en degrés de 0,5 radian, utilisez l'exemple suivant.

```
SELECT DEGREES(.5);

+-------------------+
|      degrees      |
+-------------------+
| 28.64788975654116 |
+-------------------+
```

Pour convertir PI radians en degrés, utilisez l'exemple suivant.

```
SELECT DEGREES(pi());

+---------+
| degrees |
+---------+
|     180 |
+---------+
```

Fonction DEXP

La fonction DEXP renvoie la valeur exponentielle en notation scientifique d'un nombre double précision. La seule différence entre les fonctions DEXP et EXP est que le paramètre de DEXP doit être un nombre DOUBLE PRECISION.

Syntaxe

```
DEXP(number)
```

Argument

number

Le paramètre d'entrée est un nombre `DOUBLE PRECISION`.

Type de retour

`DOUBLE PRECISION`

exemple

L'exemple suivant utilise l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Utilisez la fonction DEXP de planifier des ventes de billets selon un modèle de croissance continue. Dans cet exemple, la sous-requête renvoie le nombre de billets vendus en 2008. Ce résultat est multiplié par le résultat de la fonction DEXP, qui spécifie une croissance continue de 7 % sur 10 ans.

```
SELECT (SELECT SUM(qtysold)
FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * DEXP((7::FLOAT/100)*10) qty2010;

+-------------------+
|      qty2010      |
+-------------------+
| 695447.4837722216 |
+-------------------+
```

DLOG1 fonction

La DLOG1 fonction renvoie le logarithme naturel du paramètre d'entrée. Synonyme de Fonction LN.

DLOG10 fonction

Le DLOG1 0 renvoie le logarithme en base 10 du paramètre d'entrée.

Synonyme de Fonction LOG.

Syntaxe

```
DLOG10(number)
```

Argument

number

Le paramètre d'entrée est un nombre `DOUBLE PRECISION`.

Type de retour

`DOUBLE PRECISION`

exemple

Pour renvoyer le logarithme de base 10 du nombre 100, utilisez l'exemple suivant.

```
SELECT DLOG10(100);

+--------+
| dlog10 |
+--------+
|      2 |
+--------+
```

Fonction EXP

La fonction EXP implémente la fonction exponentielle pour une expression numérique, ou la base du logarithme naturel, e, élevée à la puissance de l'expression. La fonction EXP est l'inverse de Fonction LN.

Syntaxe

```
EXP(expression)
```

Argument

expression

L'expression doit être un type de données `INTEGER`, `DECIMAL` ou `DOUBLE PRECISION`.

Type de retour

`DOUBLE PRECISION`

exemple

L'exemple suivant utilise l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Utilisez la fonction EXP de planifier des ventes de billets selon un modèle de croissance continue. Dans cet exemple, la sous-requête renvoie le nombre de billets vendus en 2008. Ce résultat est multiplié par le résultat de la fonction EXP, qui spécifie une croissance continue de 7 % sur 10 ans.

```
SELECT (SELECT SUM(qtysold)
FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * EXP((7::FLOAT/100)*10) qty2018;

+-------------------+
|      qty2018      |
+-------------------+
| 695447.4837722216 |
+-------------------+
```

Fonction FLOOR

La fonction FLOOR arrondit un nombre au nombre entier inférieur suivant.

Syntaxe

```
FLOOR(number)
```

Argument

number

Nombre ou expression ayant pour valeur un nombre. Il peut être de type `SMALLINT`, `INTEGER`, `BIGINT`, `DECIMAL`, `FLOAT4`, `FLOAT8` ou `SUPER`.

Type de retour

FLOOR renvoie le même type de données que sont argument.

Quand l'entrée est de type SUPER, la sortie conserve le même type dynamique que l'entrée, tandis que le type statique reste le type SUPER. Quand le type dynamique de SUPER n'est pas un nombre, Amazon Redshift renvoie NULL.

Exemples

Les exemples suivants utilisent l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour montrer la valeur de la commission payée pour une transaction de vente donnée avant et après l'utilisation de la fonction FLOOR, utilisez l'exemple suivant.

```
SELECT commission
FROM sales
WHERE salesid=10000;

+------------+
| commission |
+------------+
|      28.05 |
+------------+

SELECT FLOOR(commission)
FROM sales
WHERE salesid=10000;

+-------+
| floor |
+-------+
|    28 |
+-------+
```

Fonction LN

Renvoie le logarithme naturel du paramètre d'entrée.

Synonyme de DLOG1 fonction.

Syntaxe

```
LN(expression)
```

Argument

expression

Colonne cible ou expression sur laquelle la fonction opère.

> **Note**
>
> Cette fonction renvoie une erreur pour certains types de données, si l'expression fait référence à une table créée par l'utilisateur Amazon Redshift ou à une table système STL ou STV Amazon Redshift.

Les expressions régulières avec les types de données suivants génèrent une erreur si elles font référence à une table créée par l'utilisateur ou à une table système. Les expressions régulières avec ces types de données s'exécutent exclusivement sur le nœud principal :

- `BOOLEAN`
- `CHAR`
- `DATE`
- `DECIMAL` ou `NUMERIC`
- `TIMESTAMP`
- `VARCHAR`

Les expressions régulières avec les types de données suivants s'exécutent avec succès sur des tables créées par l'utilisateur ou des tables système STL ou STV :

- `BIGINT`
- `DOUBLE PRECISION`
- `INTEGER`
- `REAL`
- `SMALLINT`

Type de retour

La fonction LN renvoie le même type que celui de l'expression en entrée.

Exemples

Pour renvoyer le logarithme naturel, ou logarithme de base e, du nombre 2,718281828, utilisez l'exemple suivant.

```
SELECT LN(2.718281828);

+--------------------+
|         ln         |
+--------------------+
| 0.9999999998311267 |
+--------------------+
```

Notez que la réponse est presque égale à 1.

L'exemple suivant utilise l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour renvoyer le logarithme naturel des valeurs de la colonne userid de la table USERS, utilisez l'exemple suivant.

```
SELECT username, LN(userid) FROM users ORDER BY userid LIMIT 10;

+----------+--------------------+
| username |         ln         |
+----------+--------------------+
| JSG99FHE |                  0 |
| PGL08LJI | 0.6931471805599453 |
| IFT66TXU | 1.0986122886681098 |
| XDZ38RDD | 1.3862943611198906 |
| AEB55QTM | 1.6094379124341003 |
| NDQ15VBM |  1.791759469228055 |
| OWY35QYB | 1.9459101490553132 |
| AZG78YIP | 2.0794415416798357 |
| MSD36KVR | 2.1972245773362196 |
| WKW41AIW |  2.302585092994046 |
+----------+--------------------+
```

Fonction LOG

Renvoie le logarithme d'un nombre.

Si vous utilisez cette fonction pour calculer le logarithme de base 10, vous pouvez également utiliser DLOG10 fonction.

Syntaxe

```
LOG([base, ]argument)
```

Paramètres

base

(Facultatif) Base de la fonction logarithmique. Ce nombre doit être positif et ne peut pas être égal à 1. Si ce paramètre est omis, Amazon Redshift calcule le logarithme de base 10 de l'argument.

argument

Argument de la fonction logarithmique. Ce nombre doit être positif. Si la valeur argument est 1, la fonction retourne 0.

Type de retour

La fonction LOG renvoie un nombre DOUBLE PRECISION.

Exemples

Pour déterminer le logarithme de base 2 du nombre 100, utilisez l'exemple suivant.

```
SELECT LOG(2, 100);
+-------------------+
|        log        |
+-------------------+
| 6.643856189774725 |
+-------------------+
```

Pour déterminer le logarithme de base 10 du nombre 100, utilisez l'exemple suivant. Notez que si vous omettez le paramètre base, Amazon Redshift suppose une base de 10.

```
SELECT LOG(100);

+-----+
| log |
+-----+
|   2 |
```

```
+-----+
```

Fonction MOD

Renvoie le reste de deux nombres, autrement dit une opération modulo. Pour calculer le résultat, le premier paramètre est divisé par le second.

Syntaxe

```
MOD(number1, number2)
```

Arguments

number1

Le premier paramètre d'entrée est un nombre `INTEGER`, `SMALLINT`, `BIGINT` ou `DECIMAL`. Si un paramètre est de type `DECIMAL`, l'autre paramètre doit également être de type `DECIMAL`. Si un paramètre est de type `INTEGER`, l'autre paramètre peut être de type `INTEGER`, `SMALLINT` ou `BIGINT`. Les deux paramètres peuvent également être de type `SMALLINT` ou `BIGINT`, mais un paramètre ne peut pas être de type `SMALLINT` si l'autre est de type `BIGINT`.

number2

Le deuxième paramètre est un nombre `INTEGER`, `SMALLINT`, `BIGINT` ou `DECIMAL`. Les mêmes règles de type de données s'appliquent à number2 en ce qui concerne number1.

Type de retour

Le type de retour de la fonction MOD est le même type numérique que les paramètres d'entrée, si les deux paramètres d'entrée sont de même type. Toutefois, si un paramètre d'entrée est de type `INTEGER`, le type renvoyé est également `INTEGER`. Les types renvoyés valides sont `DECIMAL`, `INT`, `SMALLINT` et `BIGINT`.

Notes d'utilisation

Vous pouvez utiliser `%` comme opérateur modulo.

Exemples

Pour renvoyer le reste de la division d'un nombre par un autre, utilisez l'exemple suivant.

```
SELECT MOD(10, 4);
```

```
+-----+
| mod |
+-----+
|   2 |
+-----+
```

Pour renvoyer un résultat `DECIMAL` lorsque vous utilisez la fonction MOD, utilisez l'exemple suivant.

```
SELECT MOD(10.5, 4);

+-----+
| mod |
+-----+
| 2.5 |
+-----+
```

Pour convertir un nombre avant d'exécuter la fonction MOD, utilisez l'exemple suivant. Pour de plus amples informations, veuillez consulter Fonction CAST.

```
SELECT MOD(CAST(16.4 AS INTEGER), 5);

+-----+
| mod |
+-----+
|   1 |
+-----+
```

Pour vérifier si le premier paramètre est pair en le divisant par 2, utilisez l'exemple suivant.

```
SELECT mod(5,2) = 0 AS is_even;

+---------+
| is_even |
+---------+
| false   |
+---------+
```

Pour utiliser % comme opérateur modulo, utilisez l'exemple suivant.

```
SELECT 11 % 4 as remainder;
```

```
 +-----------+
| remainder |
+-----------+
|         3 |
+-----------+
```

L'exemple suivant utilise l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour renvoyer des informations pour les catégories impaires dans la table CATEGORY, utilisez l'exemple suivant.

```
SELECT catid, catname
FROM category
WHERE MOD(catid,2)=1
ORDER BY 1,2;

+-------+-----------+
| catid |  catname  |
+-------+-----------+
|     1 | MLB       |
|     3 | NFL       |
|     5 | MLS       |
|     7 | Plays     |
|     9 | Pop       |
|    11 | Classical |
+-------+-----------+
```

Fonction PI

La fonction PI renvoie la valeur de pi à 14 décimales.

Syntaxe

```
PI()
```

Type de retour

`DOUBLE PRECISION`

Exemples

Pour renvoyer la valeur de pi, utilisez l'exemple suivant.

```
SELECT PI();

+------------------+
|        pi        |
+------------------+
| 3.141592653589793 |
+------------------+
```

Fonction POWER

La fonction POWER est une fonction exponentielle qui élève une expression numérique à la puissance d'une seconde expression numérique. Par exemple, 2 à la puissance 3 est calculé sous la forme `POWER(2,3)`, avec un résultat de 8.

Syntaxe

```
{POW | POWER}(expression1, expression2)
```

Arguments

expression1

Expression numérique à élever. Doit avoir le type de données `INTEGER`, `DECIMAL` ou `FLOAT`.

expression2

Puissance à laquelle élever expression1. Doit avoir le type de données `INTEGER`, `DECIMAL` ou `FLOAT`.

Type de retour

`DOUBLE PRECISION`

Exemples

Les exemples suivants utilisent l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Dans l'exemple suivant, la fonction POWER est utilisée pour prévoir la vente des billets des 10 prochaines années, basée sur le nombre de billets vendus en 2008 (résultat de la sous-requête). Le taux de croissance est défini sur 7 % par an dans cet exemple.

```
SELECT (SELECT SUM(qtysold) FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * POW((1+7::FLOAT/100),10) qty2010;

+-------------------+
|      qty2010      |
+-------------------+
| 679353.7540885945 |
+-------------------+
```

L'exemple suivant est une variante de l'exemple précédent, avec le taux de croissance à 7 % par an, mais l'intervalle est défini sur les mois (120 mois sur 10 ans).

```
SELECT (SELECT SUM(qtysold) FROM sales, date
WHERE sales.dateid=date.dateid
AND year=2008) * POW((1+7::FLOAT/100/12),120) qty2010;

+-----------------+
|     qty2010     |
+-----------------+
| 694034.54678046 |
+-----------------+
```

Fonction RADIANS

La fonction RADIANS convertit un angle en degrés en son équivalent en radians.

Syntaxe

```
RADIANS(number)
```

Argument

number

Le paramètre d'entrée est un nombre DOUBLE PRECISION.

Type de retour

DOUBLE PRECISION

Exemples

Pour renvoyer l'équivalent en radians de 180 degrés, utilisez l'exemple suivant.

```
SELECT RADIANS(180);

+-------------------+
|      radians      |
+-------------------+
| 3.141592653589793 |
+-------------------+
```

Fonction RANDOM

La fonction RANDOM génère une valeur aléatoire compris entre 0,0 (inclus) et 1,0 (exclusif).

Syntaxe

```
RANDOM()
```

Type de retour

`DOUBLE PRECISION`

Notes d'utilisation

Appelez RANDOM après avoir défini une valeur initiale avec la commande SET afin que RANDOM génère des nombres dans une séquence prévisible.

Exemples

Pour calculer une valeur aléatoire comprise entre 0 et 99, utilisez l'exemple suivant. Si le nombre aléatoire est compris entre 0 et 1, cette requête génère un nombre aléatoire compris entre 0 et 100.

```
SELECT CAST(RANDOM() * 100 AS INT);

+------+
| int4 |
+------+
|   59 |
+------+
```

Cet exemple utilise la commande SET pour définir une valeur SEED afin que RANDOM génère une séquence prévisible de nombres.

Pour renvoyer trois entiers RANDOM sans définir la valeur SEED, utilisez l'exemple suivant.

```
SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|    6 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   68 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   56 |
+------+
```

Pour définir la valeur SEED sur `.25` et renvoyer trois nombres RANDOM supplémentaires, utilisez l'exemple suivant.

```
SET SEED TO .25;
SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   21 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   79 |
+------+
```

```
SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   12 |
+------+
```

Pour réinitialiser la valeur SEED sur `.25` et vérifier que RANDOM renvoie les mêmes résultats que les trois appels précédents, utilisez l'exemple suivant.

```
SET SEED TO .25;
SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   21 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   79 |
+------+

SELECT CAST(RANDOM() * 100 AS INT);
+------+
| int4 |
+------+
|   12 |
+------+
```

Les exemples suivants utilisent l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour extraire un échantillon aléatoire uniforme de 10 éléments de la table SALES, utilisez l'exemple suivant.

```
SELECT *
FROM sales
ORDER BY RANDOM()
```

```
LIMIT 10;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|   45422 |  51114 |     5983 |   24482 |    4369 |   2118 |       1 |       195 |
 29.25 | 2008-10-19 05:20:07 |
|   42481 |  47638 |     4573 |    6198 |    6479 |   1987 |       4 |      1140 |
   171 | 2008-06-10 09:39:19 |
|   31494 |  34759 |    18895 |    4719 |    7753 |   2090 |       4 |      1024 |
 153.6 | 2008-09-21 03:44:26 |
|  119388 | 136685 |    21815 |   41905 |    2071 |   1884 |       1 |       359 |
 53.85 | 2008-02-27 10:43:10 |
|  166990 | 225037 |    18529 |    7628 |     746 |   2113 |       1 |      2009 |
 301.35 | 2008-10-14 10:07:44 |
|   11146 |  12096 |    42685 |    6619 |    1876 |   2123 |       1 |        29 |
  4.35 | 2008-10-24 06:23:54 |
|  148537 | 172056 |    15102 |   11787 |    6122 |   1923 |       2 |       480 |
    72 | 2008-04-07 03:58:23 |
|   68945 |  78387 |     7359 |   18323 |    6636 |   1910 |       1 |       457 |
 68.55 | 2008-03-25 08:31:03 |
|   52796 |  59576 |     9909 |   15102 |    7958 |   1951 |       1 |       479 |
 71.85 | 2008-05-05 02:25:08 |
|   90684 | 103522 |    38052 |   21549 |    7384 |   2117 |       1 |       313 |
 46.95 | 2008-10-18 05:43:11 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

Pour récupérer un échantillon aléatoire de 10 éléments tout en choisissant les éléments en fonction de leur prix, utilisez l'exemple suivant. Par exemple, un élément dont le prix est le double d'un autre a deux fois plus de chance d'apparaître dans les résultats de la requête.

```
SELECT *
FROM sales
ORDER BY -LOG(RANDOM()) / pricepaid
LIMIT 10;

+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

```
| salesid | listid | sellerid | buyerid | eventid | dateid | qtysold | pricepaid |
 commission |      saletime       |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
|  158340 | 208208 |    17082 |   42018 |    1211 |   2160 |       4 |      6852 |
 1027.8 | 2008-11-30 12:21:43 |
|   53250 |  60069 |    12644 |    7066 |    7942 |   1838 |       4 |      1528 |
 229.2 | 2008-01-12 11:24:56 |
|   22929 |  24938 |    47314 |    6503 |     179 |   2000 |       3 |       741 |
 111.15 | 2008-06-23 08:04:50 |
|  164980 | 221181 |     1949 |   19670 |    1471 |   1906 |       1 |      1330 |
 199.5 | 2008-03-21 07:59:51 |
|  159641 | 211179 |    44897 |   16652 |    7458 |   2128 |       1 |      1019 |
 152.85 | 2008-10-29 02:02:15 |
|   73143 |  83439 |     5716 |    5727 |    7314 |   1903 |       1 |       248 |
  37.2 | 2008-03-18 11:07:42 |
|   84778 |  96749 |    46608 |   32980 |    3883 |   1999 |       2 |       958 |
 143.7 | 2008-06-22 12:13:31 |
|  171096 | 232929 |    43683 |    8536 |    8353 |   1870 |       1 |       929 |
 139.35 | 2008-02-13 01:36:36 |
|   74212 |  84697 |    39809 |   15569 |    5525 |   2105 |       2 |       896 |
 134.4 | 2008-10-06 11:47:50 |
|  158011 | 207556 |    25399 |   16881 |     232 |   2088 |       2 |      2526 |
 378.9 | 2008-09-19 06:00:26 |
+---------+--------+----------+---------+---------+--------+---------+-----------
+------------+---------------------+
```

Fonction ROUND

La fonction ROUND arrondit des nombres à l'entier ou à la décimale la plus proche.

La fonction ROUND peut éventuellement inclure un second argument sous forme d'`INTEGER` pour indiquer le nombre de décimales pour l'arrondi, dans les deux sens. Lorsque vous ne fournissez pas le second argument, la fonction arrondit au nombre entier le plus proche. Lorsque le second argument integer est spécifié, la fonction arrondit au nombre le plus proche avec une précision de integer décimales.

Syntaxe

```
ROUND(number [ , integer ] )
```

Arguments

number

Nombre ou expression ayant pour valeur un nombre. Il peut être de type DECIMAL, FLOAT8 ou SUPER. Amazon Redshift peut convertir implicitement d'autres types de données numériques.

integer

(Facultatif) INTEGER qui indique le nombre de décimales pour l'arrondi dans les deux sens. Le type de données SUPER n'est pas pris en charge pour cet argument.

Type de retour

ROUND renvoie le même type de données numérique que celui du nombre en entrée.

Quand l'entrée est de type SUPER, la sortie conserve le même type dynamique que l'entrée, tandis que le type statique reste le type SUPER. Quand le type dynamique de SUPER n'est pas un nombre, Amazon Redshift renvoie NULL.

Exemples

Les exemples suivants utilisent l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour arrondir la commission payée pour une transaction donnée au nombre entier le plus proche, utilisez l'exemple suivant.

```
SELECT commission, ROUND(commission)
FROM sales WHERE salesid=10000;

+------------+-------+
| commission | round |
+------------+-------+
|      28.05 |    28 |
+------------+-------+
```

Pour arrondir la commission payée pour une transaction donnée à la première décimale, utilisez l'exemple suivant.

```
SELECT commission, ROUND(commission, 1)
FROM sales WHERE salesid=10000;
```

```
+------------+-------+
| commission | round |
+------------+-------+
|      28.05 |  28.1 |
+------------+-------+
```

Pour étendre la précision dans la direction opposée à celle de l'exemple précédent, utilisez l'exemple suivant.

```
SELECT commission, ROUND(commission, -1)
FROM sales WHERE salesid=10000;

+------------+-------+
| commission | round |
+------------+-------+
|      28.05 |    30 |
+------------+-------+
```

Fonction SIN

SIN est une fonction trigonométrique qui renvoie le sinus d'un nombre. La valeur renvoyée est comprise entre `-1` et `1`.

Syntaxe

```
SIN(number)
```

Argument

number

Nombre `DOUBLE PRECISION` en radians.

Type de retour

`DOUBLE PRECISION`

Exemples

Pour renvoyer le sinus de `-PI`, utilisez l'exemple suivant.

```
SELECT SIN(-PI());

+-------------------------+
|           sin           |
+-------------------------+
| -0.00000000000000012246 |
+-------------------------+
```

Fonction SIGN

La fonction SIGN renvoie le signe (positif ou négatif) d'un nombre. Le résultat de la fonction SIGN est 1 si l'argument est positif, `-1` si l'argument est négatif ou `0` si l'argument est `0`.

Syntaxe

```
SIGN(number)
```

Argument

number

Nombre ou expression ayant pour valeur un nombre. Il peut être de type `DECIMAL`, `FLOAT8` ou `SUPER`. D'autres types de données peuvent être convertis par Amazon Redshift via les règles de conversion implicites.

Type de retour

SIGN renvoie le même type de données numérique que celui de l'argument en entrée. Si l'entrée est de type `DECIMAL`, le résultat est de type `DECIMAL(1,0)`.

Quand l'entrée est de type `SUPER`, la sortie conserve le même type dynamique que l'entrée, tandis que le type statique reste le type `SUPER`. Lorsque le type dynamique de `SUPER` n'est pas un nombre, Amazon Redshift renvoie `NULL`.

Exemples

L'exemple suivant montre que la colonne d dans la table t2 est de type `DOUBLE PRECISION` puisque l'entrée est de type `DOUBLE PRECISION` et que la colonne n dans la table t2 a `NUMERIC(1,0)` comme sortie puisque l'entrée est de type `NUMERIC`.

```
CREATE TABLE t1(d DOUBLE PRECISION, n NUMERIC(12, 2));
INSERT INTO t1 VALUES (4.25, 4.25), (-4.25, -4.25);
CREATE TABLE t2 AS SELECT SIGN(d) AS d, SIGN(n) AS n FROM t1;
SELECT table_name, column_name, data_type FROM SVV_REDSHIFT_COLUMNS WHERE
 table_name='t1' OR table_name='t2';

+------------+-------------+----------------------+
| table_name | column_name |       data_type      |
+------------+-------------+----------------------+
| t1         | d           | double precision     |
| t1         | n           | numeric(12,2)        |
| t2         | d           | double precision     |
| t2         | n           | numeric(1,0)         |
| t1         | col1        | character varying(20) |
+------------+-------------+----------------------+
```

L'exemple suivant utilise l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour déterminer le signe de la commission payée pour une transaction donnée à partir de la table SALES, utilisez l'exemple suivant.

```
SELECT commission, SIGN(commission)
FROM sales WHERE salesid=10000;

+------------+------+
| commission | sign |
+------------+------+
|      28.05 |    1 |
+------------+------+
```

Fonction SQRT

La fonction SQRT renvoie la racine carrée d'une valeur `NUMERIC`. La racine carrée est un nombre multiplié par lui-même pour obtenir la valeur donnée.

Syntaxe

```
SQRT(expression)
```

Argument

expression

L'expression doit avoir un type de données `INTEGER`, `DECIMAL` ou `FLOAT`, ou un type de données qui est implicitement converti vers ces types de données. L'expression peut inclure des fonctions.

Type de retour

`DOUBLE PRECISION`

Exemples

Pour renvoyer la racine carrée de 16, utilisez l'exemple suivant.

```
SELECT SQRT(16);

+------+
| sqrt |
+------+
|    4 |
+------+
```

Pour renvoyer la racine carrée de la chaîne 16 en utilisant une conversion de type implicite, utilisez l'exemple suivant.

```
SELECT SQRT('16');

+------+
| sqrt |
+------+
|    4 |
+------+
```

Pour renvoyer la racine carrée de 16,4 après avoir utilisé la fonction ROUND, utilisez l'exemple suivant.

```
SELECT SQRT(ROUND(16.4));
```

```
+------+
| sqrt |
+------+
|    4 |
+------+
```

Pour renvoyer le rayon lorsque l'aire du cercle est donnée, utilisez l'exemple suivant. Il calcule le rayon en pouces, par exemple, lorsque la surface est indiquée en pouces carrés. Dans l'exemple, l'aire est de 20.

```
SELECT SQRT(20/PI()) AS radius;

+--------------------+
|      radius        |
+--------------------+
| 2.5231325220201604 |
+--------------------+
```

Les exemples suivants utilisent l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour renvoyer la racine carrée des valeurs de COMMISSION de la table SALES, utilisez l'exemple suivant. La colonne COMMISSION est une colonne `DECIMAL`. Cet exemple montre comment utiliser la fonction dans une requête ayant une logique conditionnelle plus complexe.

```
SELECT SQRT(commission)
FROM sales WHERE salesid < 10 ORDER BY salesid;

+--------------------+
|        sqrt        |
+--------------------+
| 10.449880382090505 |
| 3.3763886032268267 |
|  7.245688373094719 |
|  5.123475382979799 |
|  4.806245936279167 |
|  7.687652437513028 |
| 10.871982339941507 |
| 5.4359911699707535 |
|   9.41541289588513 |
+--------------------+
```

Pour renvoyer la racine carré arrondie du même ensemble de valeurs COMMISSION, utilisez l'exemple suivant.

```
SELECT ROUND(SQRT(commission))
FROM sales WHERE salesid < 10 ORDER BY salesid;

+-------+
| round |
+-------+
|    10 |
|     3 |
|     7 |
|     5 |
|     5 |
|     8 |
|    11 |
|     5 |
|     9 |
+-------+
```

Fonction TAN

TAN est une fonction trigonométrique qui renvoie la tangente d'un nombre. L'argument d'entrée est un nombre (en radians).

Syntaxe

```
TAN(number)
```

Argument

number

Nombre DOUBLE PRECISION.

Type de retour

DOUBLE PRECISION

Exemples

Pour renvoyer la tangente de zéro, utilisez l'exemple suivant.

```
SELECT TAN(0);

+-----+
| tan |
+-----+
|   0 |
+-----+
```

Fonction TRUNC

La fonction TRUNC tronque les nombres à l'entier ou à la décimale précédente.

La fonction TRUNC peut éventuellement inclure un second argument sous forme d'`INTEGER` pour indiquer le nombre de décimales pour l'arrondi, dans les deux sens. Lorsque vous ne fournissez pas le second argument, la fonction arrondit au nombre entier le plus proche. Lorsque le second argument integer est spécifié, la fonction arrondit au nombre le plus proche avec une précision de integer décimales.

Cette fonction peut également tronquer une valeur `TIMESTAMP` et renvoyer une valeur `DATE`. Pour de plus amples informations, veuillez consulter Fonction TRUNC.

Syntaxe

```
TRUNC(number [ , integer ])
```

Arguments

number

Nombre ou expression ayant pour valeur un nombre. Il peut être de type `DECIMAL`, `FLOAT8` ou `SUPER`. D'autres types de données peuvent être convertis par Amazon Redshift via les règles de conversion implicites.

integer

(Facultatif) `INTEGER` qui indique le nombre de décimales de précision, dans les deux sens. Si aucun nombre entier n'est fourni, le nombre est tronqué en tant que nombre entier ; si un nombre entier est spécifié, le nombre est tronqué à la décimale spécifiée. Ceci n'est pas pris en charge pour le type de données `SUPER`.

Type de retour

TRUNC renvoie le même type de données numérique que celui du nombre en entrée.

Quand l'entrée est de type SUPER, la sortie conserve le même type dynamique que l'entrée, tandis que le type statique reste le type SUPER. Quand le type dynamique de SUPER n'est pas un nombre, Amazon Redshift renvoie NULL.

Exemples

Certains des exemples suivants utilisent l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour tronquer la commission payée pour une transaction de vente donnée, utilisez l'exemple suivant.

```
SELECT commission, TRUNC(commission)
FROM sales WHERE salesid=784;

+------------+-------+
| commission | trunc |
+------------+-------+
|     111.15 |   111 |
+------------+-------+
```

Pour tronquer la même valeur de commission que la première décimale, utilisez l'exemple suivant.

```
SELECT commission, TRUNC(commission,1)
FROM sales WHERE salesid=784;

+------------+-------+
| commission | trunc |
+------------+-------+
|     111.15 | 111.1 |
+------------+-------+
```

Pour tronquer la commission avec une valeur négative pour le deuxième argument, utilisez l'exemple suivant. Notez que 111.15 est arrondi vers le bas à 110.

```
SELECT commission, TRUNC(commission,-1)
FROM sales WHERE salesid=784;

+------------+-------+
```

```
| commission | trunc |
+------------+-------+
|     111.15 |   110 |
+------------+-------+
```

Fonctions d’objet

Voici les fonctions d’objet SQL prises en charge par Amazon Redshift pour créer des objets de type SUPER :

Rubriques

- Fonction LOWER_ATTRIBUTE_NAMES
- Fonction OBJECT
- Fonction OBJECT_TRANSFORM
- Fonction UPPER_ATTRIBUTE_NAMES

Fonction LOWER_ATTRIBUTE_NAMES

Convertit tous les noms d'attributs applicables dans une valeur SUPER en minuscules, en utilisant la même routine de conversion majuscules/minuscules que le. Fonction LOWER LOWER_ATTRIBUTE_NAMES prend en charge les caractères multi-octets UTF-8, jusqu'à un maximum de quatre octets par caractère.

Pour convertir les noms d'attributs SUPER en majuscules, utilisez leFonction UPPER_ATTRIBUTE_NAMES.

Syntaxe

```
LOWER_ATTRIBUTE_NAMES(super_expression)
```

Arguments

super_expression

Une super expression.

Type de retour

SUPER

Notes d’utilisation

Dans Amazon Redshift, les identifiants de colonne ne font généralement pas la distinction entre majuscules et minuscules et sont convertis en minuscules. Si vous ingérez des données provenant de formats de données sensibles aux majuscules, tels que JSON, les données peuvent contenir des noms d'attributs composés de majuscules et de minuscules.

Prenez l’exemple de code suivant.

```
CREATE TABLE t1 (s) AS SELECT JSON_PARSE('{"AttributeName": "Value"}');


SELECT s.AttributeName FROM t1;

attributename
-------------
NULL


SELECT s."AttributeName" FROM t1;

attributename
-------------
NULL
```

Amazon Redshift renvoie la valeur NULL pour les deux requêtes. Pour effectuer une requête`AttributeName`, utilisez LOWER_ATTRIBUTE_NAMES pour convertir les noms des attributs des données en minuscules. Prenez l’exemple de code suivant.

```
CREATE TABLE t2 (s) AS SELECT LOWER_ATTRIBUTE_NAMES(s) FROM t1;


SELECT s.attributename FROM t2;

attributename
-------------
"Value"


SELECT s.AttributeName FROM t2;

attributename
```

```
-------------
"Value"


SELECT s."attributename" FROM t2;

attributename
-------------
"Value"


SELECT s."AttributeName" FROM t2;

attributename
-------------
"Value"
```

L'option de `enable_case_sensitive_super_attribute` configuration est une option connexe pour travailler avec des noms d'attributs d'objets en majuscules et minuscules. Elle permet à Amazon Redshift de reconnaître les majuscules dans les noms d'attributs SUPER. Cela peut être une solution alternative à l'utilisation de LOWER_ATTRIBUTE_NAMES. Pour plus d'informations sur`enable_case_sensitive_super_attribute`, rendez-vous sur[enable_case_sensitive_super_attribute](#).

Exemples

Conversion des noms d'attributs SUPER en minuscules

L'exemple suivant utilise LOWER_ATTRIBUTE_NAMES pour convertir les noms d'attribut de toutes les valeurs SUPER d'une table.

```
-- Create a table and insert several SUPER values.
CREATE TABLE t (i INT, s SUPER);

INSERT INTO t VALUES
  (1, NULL),
  (2, 'A'::SUPER),
  (3, JSON_PARSE('{"AttributeName": "B"}')),
  (4, JSON_PARSE(
     '[{"Subobject": {"C": "C"},
       "Subarray": [{"D": "D"}, "E"]
      }]'));
```

```
-- Convert all attribute names to lowercase.
UPDATE t SET s = LOWER_ATTRIBUTE_NAMES(s);

SELECT i, s FROM t ORDER BY i;

 i |                         s
---+---------------------------------------------------
 1 | NULL
 2 | "A"
 3 | {"attributename":"B"}
 4 | [{"subobject":{"c":"C"},"subarray":[{"d":"D"}, "E"]}]
```

Observez le fonctionnement de LOWER_ATTRIBUTE_NAMES.

- Les valeurs NULL et les valeurs scalaires SUPER telles que celles-ci restent `"A"` inchangées.
- Dans un objet SUPER, tous les noms d'attributs sont remplacés par des minuscules, mais leurs valeurs `"B"` restent inchangées.
- LOWER_ATTRIBUTE_NAMES s'applique de manière récursive à tout objet SUPER imbriqué dans un tableau SUPER ou dans un autre objet.

Utilisation de LOWER_ATTRIBUTE_NAMES sur un objet SUPER avec des noms d'attributs dupliqués

Si un objet SUPER contient des attributs dont les noms ne diffèrent que dans leur cas, LOWER_ATTRIBUTE_NAMES générera une erreur. Prenez l'exemple de code suivant.

```
SELECT LOWER_ATTRIBUTE_NAMES(JSON_PARSE('{"A": "A", "a": "a"}'));

error:   Invalid input
code:    8001
context: SUPER value has duplicate attributes after case conversion.
```

Fonction OBJECT

Crée un tableau du type de données SUPER.

Syntaxe

```
OBJECT ( [ key1, value1 ], [ key2, value2 ...] )
```

Arguments

key1, key2 (clé 1, clé 2)

Expressions qui évaluent des chaînes de type VARCHAR.

value1, value2

Expressions de n'importe quel type de données Amazon Redshift à l'exception des types datetime, car Amazon Redshift ne convertit pas les types datetime en type de données SUPER. Pour obtenir plus d'informations sur les types datetime, consultez Types datetime.

Les expressions `value` dans un objet ne doivent pas nécessairement appartenir au même type de données.

Type de retour

SUPER

exemple

```
-- Creates an empty object.
select object();

object
--------
{}
(1 row)

-- Creates objects with different keys and values.
select object('a', 1, 'b', true, 'c', 3.14);

object
---------------------------
{"a":1,"b":true,"c":3.14}
(1 row)

select object('a', object('aa', 1), 'b', array(2,3), 'c', json_parse('{}'));

object
---------------------------------
{"a":{"aa":1},"b":[2,3],"c":{}}
(1 row)
```

```
-- Creates objects using columns from a table.
create table bar (k varchar, v super);
insert into bar values ('k1', json_parse('[1]')), ('k2', json_parse('{}'));
select object(k, v) from bar;

object
------------
{"k1":[1]}
{"k2":{}}
(2 rows)

-- Errors out because DATE type values can't be converted to SUPER type.
select object('k', '2008-12-31'::date);

ERROR:  OBJECT could not convert type date to super
```

Fonction OBJECT_TRANSFORM

Transforme un objet SUPER.

Syntaxe

```
OBJECT_TRANSFORM(
  input
  [KEEP path1, ...]
  [SET
    path1, value1,
    ...,  ...
  ]
)
```

Arguments

input

Expression qui se résout en un objet de type SUPER.

KEEP

Toutes les valeurs de chemin spécifiées dans cette clause sont conservées et transférées vers l’objet de sortie.

Cette clause est facultative.

path1, path2, ...

Litéraux de chaîne constants, au format de composants de chemin entre guillemets doubles délimités par des points. Par exemple, `'"a"."b"."c"'` est une valeur de chemin valide. Ceci s'applique au paramètre path dans les clauses KEEP et SET.

SET

paires path et value permettant de modifier un chemin existant ou d'ajouter un nouveau chemin, et définir la valeur de celui-ci dans l'objet de sortie.

Cette clause est facultative.

value1, value2, ...

Expressions qui se résolvent en valeurs de type SUPER. Notez que les types numeric, text et boolean peuvent être résolus en SUPER.

Type de retour

`SUPER`

Notes d'utilisation

OBJECT_TRANSFORM renvoie un objet de type SUPER contenant les valeurs de chemin provenant de input spécifiées dans KEEP et les paires path value spécifiées dans SET.

Si KEEP et SET sont vides, OBJECT_TRANSFORM renvoie input.

Si input n'est pas un objet (object) de type SUPER, OBJECT_TRANSFORM renvoie input, quelles que soient les valeurs KEEP ou SET.

exemple

L'exemple suivant transforme un objet SUPER en un autre objet SUPER.

```
CREATE TABLE employees (
    col_person SUPER
);

INSERT INTO employees
VALUES
    (
        json_parse('
```

```
            {
                "name": {
                    "first": "John",
                    "last": "Doe"
                },
                "age": 25,
                "ssn": "111-22-3333",
                "company": "Company Inc.",
                "country": "U.S."
            }
        ')
    ),
    (
        json_parse('
            {
                "name": {
                    "first": "Jane",
                    "last": "Appleseed"
                },
                "age": 34,
                "ssn": "444-55-7777",
                "company": "Organization Org.",
                "country": "Ukraine"
            }
        ')
    )
;

SELECT
    OBJECT_TRANSFORM(
        col_person
        KEEP
            '"name"."first"',
            '"age"',
            '"company"',
            '"country"'
        SET
            '"name"."first"', UPPER(col_person.name.first::TEXT),
            '"age"', col_person.age + 5,
            '"company"', 'Amazon'
    ) AS col_person_transformed
FROM employees;

--This result is formatted for ease of reading.
```

```
                  col_person_transformed
-------------------------------------------------------------
{
    "name": {
        "first": "JOHN"
    },
    "age": 30,
    "company": "Amazon",
    "country": "U.S."
}
{
    "name": {
        "first": "JANE"
    },
    "age": 39,
    "company": "Amazon",
    "country": "Ukraine"
}
```

Fonction UPPER_ATTRIBUTE_NAMES

Convertit tous les noms d'attributs applicables dans une valeur SUPER en majuscules, en utilisant la même routine de conversion majuscules que leFonction UPPER. UPPER_ATTRIBUTE_NAMES prend en charge les caractères multi-octets UTF-8, jusqu'à un maximum de quatre octets par caractère.

Pour convertir les noms d'attributs SUPER en minuscules, utilisez le. Fonction LOWER_ATTRIBUTE_NAMES

Syntaxe

```
UPPER_ATTRIBUTE_NAMES(super_expression)
```

Arguments

super_expression

Une super expression.

Type de retour

SUPER

Exemples

Conversion des noms d'attributs SUPER en majuscules

L'exemple suivant utilise UPPER_ATTRIBUTE_NAMES pour convertir les noms d'attribut de toutes les valeurs SUPER d'une table.

```
-- Create a table and insert several SUPER values.
CREATE TABLE t (i INT, s SUPER);

INSERT INTO t VALUES
  (1, NULL),
  (2, 'a'::SUPER),
  (3, JSON_PARSE('{"AttributeName": "b"}')),
  (4, JSON_PARSE(
     '[{"Subobject": {"c": "c"},
       "Subarray": [{"d": "d"}, "e"]
      }]'));

-- Convert all attribute names to uppercase.
UPDATE t SET s = UPPER_ATTRIBUTE_NAMES(s);

SELECT i, s FROM t ORDER BY i;

 i |                         s
---+---------------------------------------------------
 1 | NULL
 2 | "a"
 3 | {"ATTRIBUTENAME":"B"}
 4 | [{"SUBOBJECT":{"C":"c"},"SUBARRAY":[{"D":"d"}, "e"]}]
```

Observez le fonctionnement de UPPER_ATTRIBUTE_NAMES.

- Les valeurs NULL et les valeurs scalaires SUPER telles que celles-ci restent "a" inchangées.
- Dans un objet SUPER, tous les noms d'attributs sont remplacés par des majuscules, mais leurs valeurs "b" restent inchangées.
- UPPER_ATTRIBUTE_NAMES s'applique de manière récursive à tout objet SUPER imbriqué dans un tableau SUPER ou dans un autre objet.

Utilisation de UPPER_ATTRIBUTE_NAMES sur un objet SUPER avec des noms d'attributs dupliqués

Si un objet SUPER contient des attributs dont les noms ne diffèrent que dans leur cas, UPPER_ATTRIBUTE_NAMES générera une erreur. Prenez l'exemple de code suivant.

```
SELECT UPPER_ATTRIBUTE_NAMES(JSON_PARSE('{"A": "A", "a": "a"}'));

error:   Invalid input
code:    8001
context: SUPER value has duplicate attributes after case conversion.
```

Fonctions spatiales

Les relations entre les objets de géométrie sont basées sur le modèle DE-9IM (Dimensionally Extended nine-Intersection Model). Ce modèle définit des prédicats tels que est égal à, contient et couvre. Pour plus d'informations sur la définition des relations spatiales, consultez DE-9IM dans Wikipedia.

Pour plus d'informations sur l'utilisation des données spatiales avec Amazon Redshift, consultez Interrogation des données spatiales dans Amazon Redshift.

Amazon Redshift fournit des fonctions spatiales qui fonctionnent avec les types de données `GEOMETRY` et `GEOGRAPHY`. La liste suivante répertorie les fonctions qui prennent en charge le type de données `GEOGRAPHY` :

- ST_Area
- ST_AsEWKT
- ST_JSON AsGeo
- ST_WEBB AsHex
- ST_WKB AsHex
- ST_ AsText
- ST_Distance
- ST_ GeogFromText
- ST_WKB GeogFrom
- ST_Length
- ST_ NPoints
- ST_Perimeter

La liste suivante répertorie l'ensemble complet des fonctions spatiales prises en charge par Amazon Redshift.

Rubriques

- Ajouter BBox
- Lâchez BBox
- GeometryType
- H3_Boundary
- H3_Centre
- H3_ FromLongLat
- H3_ FromPoint
- H3_Polyfill
- ST_ AddPoint
- ST_Angle
- ST_Area
- ST_ AsBinary
- ST_AsEWKB
- ST_AsEWKT
- ST_JSON AsGeo
- ST_WKB AsHex
- ST_WEBB AsHex
- ST_ AsText
- ST_Azimuth
- ST_Boundary
- ST_Buffer
- ST_Centroid
- ST_Collect
- ST_Contains
- ST_ ContainsProperly
- ST_ ConvexHull

- ST_ CoveredBy
- ST_Covers
- ST_Crosses
- ST_Dimension
- ST_Disjoint
- ST_Distance
- ST_ DistanceSphere
- ST_ DWithin
- ST_ EndPoint
- ST_Enveloppe
- ST_Equals
- ST_ ExteriorRing
- ST_Force2D
- ST_Force3D
- ST_Force3DM
- ST_Force3DZ
- ST_Force4D
- ST_ GeoHash
- ST_ GeogFromText
- ST_WKB GeogFrom
- ST_GeometryN
- ST_ GeometryType
- ST_WEBB GeomFrom
- ST_EWKT GeomFrom
- ST_ GeomFromGeoHash
- ST_JSON GeomFromGeo
- ST_ GeomFromGeoSquare
- ST_ GeomFromText

- ST_WKB GeomFrom
- ST_ GeoSquare
- ST_N InteriorRing
- ST_Intersects
- ST_Intersection
- ST_CCW IsPolygon
- ST_CW IsPolygon
- ST_ IsClosed
- ST_ IsCollection
- ST_ IsEmpty
- ST_ IsRing
- ST_ IsSimple
- ST_ IsValid
- ST_Length
- ST_ LengthSphere
- ST_Length2D
- ST_ LineFromMultiPoint
- ST_ LineInterpolatePoint
- ST_M
- ST_ MakeEnvelope
- ST_ MakeLine
- ST_ MakePoint
- ST_ MakePolygon
- ST_ MemSize
- ST_ MMax
- ST_ MMin
- ST_Multi
- ST_ NDims

- ST_ NPoints
- ST_ NRings
- ST_ NumGeometries
- ST_ NumInteriorRings
- ST_ NumPoints
- ST_Perimeter
- ST_Perimeter2D
- ST_Point
- ST_PointN
- ST_Points
- ST_Polygon
- ST_ RemovePoint
- ST_Reverse
- ST_ SetPoint
- ST_SetSRID
- ST_Simplify
- ST_SRID
- ST_ StartPoint
- ST_Touches
- ST_Transform
- ST_Union
- ST_Within
- ST_X
- ST_ XMax
- ST_ XMin
- ST_Y
- ST_ YMax
- ST_ YMin
- ST_Z

- ST_ ZMax
- ST_ ZMin
- Supports BBox

Ajouter BBox

Add BBox renvoie une copie de la géométrie d'entrée qui prend en charge le codage avec un cadre de délimitation précalculé. Pour plus d'informations sur la prise en charge des cadres de délimitation, consultez Cadre de délimitation.

Syntaxe

```
AddBBox(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

GEOMETRY

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant renvoie une copie d'une géométrie de polygone en entrée qui prend en charge l'encodage avec un cadre de délimitation.

```
SELECT ST_AsText(AddBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
 st_astext
----------
 POLYGON((0 0,1 0,0 1,0 0))
```

Lâchez BBox

Drop BBox renvoie une copie de la géométrie d'entrée qui ne prend pas en charge le codage avec un cadre de délimitation précalculé. Pour plus d'informations sur la prise en charge des cadres de délimitation, consultez Cadre de délimitation.

Syntaxe

```
DropBBox(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

GEOMETRY

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant renvoie une copie d'une géométrie de polygone en entrée qui ne prend pas en charge l'encodage avec un cadre de délimitation.

```
SELECT ST_AsText(DropBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
 st_astext
----------
 POLYGON((0 0,1 0,0 1,0 0))
```

GeometryType

GeometryType renvoie le sous-type d'une géométrie d'entrée sous forme de chaîne.

Syntaxe

```
GeometryType(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

VARCHAR représentant le sous-type de geom.

Si geom est null, null est renvoyé.

Les valeurs renvoyées sont comme suit :

Valeur de chaîne renvoyée pour les géométries 2D, 3DZ, 4D	Valeur de chaîne renvoyée pour les géométries 3DM	Sous-type de géométrie
POINT	POINTM	Renvoyé si geom est un sous-type POINT
LINESTRING	LINESTRINGM	Renvoyé si geom est un sous-type LINESTRING
POLYGON	POLYGONM	Renvoyé si geom est un sous-type POLYGON
MULTIPOINT	MULTIPOINTM	Renvoyé si geom est un sous-type MULTIPOINT
MULTILINESTRING	MULTILINESTRINGM	Renvoyé si geom est un sous-type MULTILINESTRING
MULTIPOLYGON	MULTIPOLYGONM	Renvoyé si geom est un sous-type MULTIPOLYGON
GEOMETRYCOLLECTION	GEOMETRYCOLLECTIONM	Renvoyé si geom est un sous-type GEOMETRYCOLLECTION

Exemples

Le SQL suivant convertit une représentation WKT d'un polygone et renvoie le sous-type GEOMETRY sous forme de chaîne.

```
SELECT GeometryType(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
geometrytype
-------------
 POLYGON
```

H3_Boundary

H3_Boundary renvoie la limite d'un identifiant de cellule H3 à partir d'un index d'entrée. Pour en savoir plus sur l'indexation H3, consultez H3.

Syntaxe

```
H3_Boundary(index)
```

Arguments

index

Valeur de type de données BIGINT ou VARCHAR représentant l'index d'une cellule H3. Ou une expression qui correspond à l'un de ces types de données.

Type de retour

POLYGON— représente le polygone dont l'identifiant du système de référence spatiale (SRID) est de. 0

Si l'index n'est pas valide, une erreur est renvoyée.

Exemples

Le code SQL suivant saisit un VARCHAR qui représente l'index d'une cellule H3 et renvoie un POLYGONE avec un SRID 0 qui représente la limite de la cellule H3 en entrée. La sortie de H3_Boundary est entrée dans ST_Awewkt pour être affichée sous forme de texte connu étendu (EWKT).

```
SELECT ST_AsEWKT(H3_Boundary('8025fffffffffff'));
```

```
 st_asewkt
-----------------------------------------------------------------------------------------------------------
 POLYGON((87.7729641223953 52.6542030078627,86.7082098104731
 40.3127667561966,98.2285042557705 33.6210697806835,110.694610548823
 37.163896485796,116.212895637138 47.3094513028131,106.40349563788
 56.210610737585,87.7729641223953 52.6542030078627))
```

Le code SQL suivant saisit un BIGINT qui représente l'index d'une cellule H3 et renvoie un POLYGONE avec un SRID 0 qui représente la limite de la cellule H3 en entrée. La sortie de H3_Boundary est entrée dans ST_Awewkt pour être affichée sous forme de texte connu étendu (EWKT).

```
SELECT ST_AsEWKT(H3_Boundary(577129255373111295));
```

```
 st_asewkt
-----------------------------------------------------------------------------------------------------------
POLYGON((87.7729641223953 52.6542030078627,86.7082098104731
 40.3127667561966,98.2285042557705 33.6210697806835,110.694610548823
 37.163896485796,116.212895637138 47.3094513028131,106.40349563788
 56.210610737585,87.7729641223953 52.6542030078627))
```

H3_Centre

H3_Center renvoie le centroïde d'un identifiant de cellule H3 à partir d'un index d'entrée. Pour en savoir plus sur l'indexation H3, consultez H3.

Syntaxe

```
H3_Center(index)
```

Arguments

index

Valeur de type de données BIGINT ou VARCHAR représentant l'index d'une cellule H3. Ou une expression qui correspond à l'un de ces types de données.

Type de retour

`POINT`— représente le centre de gravité de la cellule H3 avec l'identifiant du système de référence spatiale (SRID) de. `0`

Si l'index n'est pas valide, une erreur est renvoyée.

Exemples

Le code SQL suivant saisit un `VARCHAR` qui représente l'index d'une cellule H3 et renvoie un POINT avec un SRID de 0 qui représente le centre de gravité de la cellule H3 en entrée.

```
SELECT H3_Center('8025fffffffffff');
```

```
 h3_center
--------------------------------------------
 010100000070707A550B605940AEE9D70B327E4640
```

Le code SQL suivant saisit un `BIGINT` qui représente l'index d'une cellule H3 et renvoie un POINT avec un SRID 0 qui représente le centre de gravité de la cellule H3 en entrée.

```
SELECT H3_Center(577129255373111295);
```

```
 h3_center
--------------------------------------------
 010100000070707A550B605940AEE9D70B327E4640
```

Le code SQL suivant saisit un `VARCHAR` qui représente l'index d'une cellule H3 et renvoie un POINT avec un SRID 0 qui représente le centre de gravité de la cellule H3 en entrée. La sortie de H3_Center est entrée dans ST_Awewkt pour être affichée sous forme de texte connu étendu (EWKT).

```
SELECT ST_AsEWKT(H3_Center('8075fffffffffff'));
```

```
 st_asewkt
----------------------------------------
```

```
POINT(-5.24539029677733 2.30088211162675)
```

H3_ FromLongLat

H3_ FromLongLat renvoie l'ID de cellule H3 correspondant à partir d'une longitude, d'une latitude et d'une résolution entrées. Pour en savoir plus sur l'indexation H3, consultez H3.

Syntaxe

```
H3_FromLongLat(longitude, lattitude, resolution)
```

Arguments

longitude

Valeur de type de données `DOUBLE PRECISION` ou expression qui est évaluée sur un type `DOUBLE PRECISION`.

latitude

Valeur de type de données `DOUBLE PRECISION` ou expression qui est évaluée sur un type `DOUBLE PRECISION`.

resolution

Valeur de type de données `INTEGER` ou expression qui est évaluée sur un type `INTEGER`. La valeur représente la résolution du système de grille H3. La valeur doit être un entier compris entre 0 et 15. `0` étant la résolution la plus grossière et `15` la plus fine.

Type de retour

`BIGINT` : représente l'identifiant de la cellule H3.

Si l'argument resolution est hors limites, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie l'ID de cellule H3 à partir de la longitude `0`, de la latitude `0` et de la résolution `10`.

```
SELECT H3_FromLongLat(0, 0, 10);
```

```
 h3_fromlonglat
-------------------
 623560421467684863
```

H3_ FromPoint

H3_ FromPoint renvoie l'ID de cellule H3 correspondant à partir d'un point de géométrie et d'une résolution en entrée. Pour en savoir plus sur l'indexation H3, consultez H3.

Syntaxe

```
H3_FromPoint(geom, resolution)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. L'argument geom doit être un POINT.

resolution

Valeur de type de données INTEGER ou expression qui est évaluée sur un type INTEGER. La valeur représente la résolution du système de grille H3. La valeur doit être un entier compris entre 0 et 15. 0 étant la résolution la plus grossière et 15 la plus fine.

Type de retour

BIGINT : représente l'identifiant de la cellule H3.

Si geom n'est pas une POINT, une erreur est renvoyée.

Si l'argument resolution est hors limites, une erreur est renvoyée.

Si geom est vide, la valeur NULL est renvoyée.

Exemples

Le code SQL suivant renvoie l'ID de cellule H3 à partir du point 0,0 et de la résolution 10.

```
SELECT H3_FromPoint(ST_GeomFromText('POINT(0 0)'), 10);
```

```
 h3_frompoint
-------------------
 623560421467684863
```

H3_Polyfill

H3_Polyfill renvoie la cellule H3 correspondante IDs qui correspond aux hexagones et aux pentagones contenus dans le polygone d'entrée de la résolution donnée. Pour en savoir plus sur l'indexation H3, consultez H3.

Syntaxe

```
H3_Polyfill(geom, resolution)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. L'argument geom doit être un POLYGON.

resolution

Valeur de type de données INTEGER ou expression qui est évaluée sur un type INTEGER. La valeur représente la résolution du système de grille H3. La valeur doit être un entier compris entre 0 et 15. 0 étant la résolution la plus grossière et 15 la plus fine.

Type de retour

SUPER— représente une liste de cellules IDs H3.

Si geom n'est pas une POLYGON, une erreur est renvoyée.

Si l'argument resolution est hors limites, une erreur est renvoyée.

Si geom est vide, la valeur NULL est renvoyée.

Exemples

Le code SQL suivant renvoie un tableau de types de données SUPER composé de cellules H3 IDs à partir d'un polygone et d'une résolution. 4

```
SELECT H3_Polyfill(ST_GeomFromText('POLYGON((0 0, 0 1, 1 1, 1 0, 0 0))'), 4);
```

```
 h3_polyfill
-------------------------------------------------------------------------------------------------
 [596538848238895103,596538805289222143,596538856828829695,596538813879156735,5965379205259591
```

ST_ AddPoint

ST_ AddPoint renvoie une géométrie de chaîne de lignes identique à la géométrie d'entrée avec un point ajouté. Si un index est fourni, le point est ajouté à la position de l'index. Si l'index est -1 ou non fourni, le point est ajouté à linestring.

L'index est basé sur zéro. L'identificateur du système de référence spatiale (SRID) du résultat est identique à celui de la géométrie en entrée.

La dimension de la géométrie renvoyée est identique à celle de la valeur geom1. Si geom1 et geom2 ont des dimensions différentes, geom2 est projeté à la dimension de geom1.

Syntaxe

```
ST_AddPoint(geom1, geom2)
```

```
ST_AddPoint(geom1, geom2, index)
```

Arguments

geom1

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type doit être `LINESTRING`.

geom2

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type doit être `POINT`. Le point peut être le point vide.

index

Valeur de type de données INTEGER qui représente la position d'un index de base zéro.

Type de retour

GEOMETRY

Si geom1, geom2, ou index est null, alors null est renvoyé.

Si geom2 est le point vide, une copie de geom1 est renvoyée.

Si geom1 n'est pas un LINESTRING, une erreur est renvoyée.

Si geom2 n'est pas un POINT, une erreur est renvoyée.

Si l'index est hors de portée, une erreur est renvoyée. Les valeurs valides pour la position d'index sont -1 ou une valeur comprise entre 0 et ST_NumPoints(geom1).

Exemples

Le code SQL suivant ajoute un point à un linestring pour en faire une linestring fermée.

```
WITH tmp(g) AS (SELECT ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5)',4326))
SELECT ST_AsEWKT(ST_AddPoint(g, ST_StartPoint(g))) FROM tmp;
```

```
 st_asewkt
-------------------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 5,0 5,0 0)
```

Le code SQL suivant ajoute un point à une position spécifique dans une linestring.

```
WITH tmp(g) AS (SELECT ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5)',4326))
SELECT ST_AsEWKT(ST_AddPoint(g, ST_SetSRID(ST_Point(5, 10), 4326), 3)) FROM tmp;
```

```
 st_asewkt
-------------------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 10,5 5,0 5)
```

ST_Angle

ST_Angle renvoie l'angle en radians entre les points mesurés dans le sens des aiguilles d'une montre comme suit :

- Si trois points sont entrés, l'angle P1-P2-P3 renvoyé est mesuré comme si l'angle était obtenu en tournant de P1 à P3 autour de P2 dans le sens des aiguilles d'une montre.
- Si quatre points sont entrés, l'angle dans le sens des aiguilles d'une montre renvoyé et formé par les lignes orientées P1-P2 et P3-P4 est renvoyé. Si l'entrée est un cas dégénéré (c'est-à-dire que P1 est égal à P2 ou que P3 égal à P4), la valeur null est renvoyée.

La valeur de renvoi est en radians et dans la plage [0, 2π).

ST_Angle fonctionne sur des projections 2D des géométries en entrée.

Syntaxe

```
ST_Angle(geom1, geom2, geom3)
```

```
ST_Angle(geom1, geom2, geom3, geom4)
```

Arguments

geom1

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type doit être `POINT`.

geom2

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type doit être `POINT`.

geom3

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type doit être `POINT`.

geom4

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type doit être `POINT`.

Type de retour

`DOUBLE PRECISION`.

Si geom1 est égal à geom2 ou geom2 est égal à geom3, alors une valeur null est renvoyée.

Si geom1, geom2, geom3, ou geom4 est null, alors une valeur null est renvoyée.

Si geom1, geom2, geom3 ou geom4 est le point vide, une erreur est renvoyée.

Si geom1, geom2, geom3 et geom4 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Exemples

Le SQL suivant renvoie l'angle converti en degrés de trois points d'entrée.

```
SELECT ST_Angle(ST_Point(1,1), ST_Point(0,0), ST_Point(1,0)) / Pi() * 180.0 AS angle;
```

```
 angle
---------------
    45
```

Le SQL suivant renvoie l'angle converti en degrés de quatre points d'entrée.

```
SELECT ST_Angle(ST_Point(1,1), ST_Point(0,0), ST_Point(1,0), ST_Point(2,0)) / Pi() *
 180.0 AS angle;
```

```
 angle
---------------
   225
```

ST_Area

Pour une géographie d'entrée, ST_Area renvoie la zone cartésienne de la projection 2D. Les unités de surface sont les mêmes que les unités dans lesquelles les coordonnées de la géométrie en entrée sont exprimées. Pour les points, les linestrings, les multipoints et les multilinestrings, la fonction

renvoie 0. Pour les collections de géométrie, elle renvoie la somme des zones des géométries de la collection.

Pour une géographie d'entrée, ST_Area renvoie la zone géodésique de la projection 2D d'une géographie surfacique en entrée calculée sur le sphéroïde déterminé par le SRID. L'unité de longueur est en mètres carrés. La fonction renvoie zéro (0) pour les points, les multipoints et les géomgraphies linéaires. Lorsque l'entrée est une collection de géométries, la fonction renvoie la somme des zones des géographies surfaciques de la collection.

Syntaxe

```
ST_Area(geo)
```

Arguments

geo

Valeur de type de données GEOMETRY ou GEOGRAPHY ou expression qui est évaluée sur un type GEOMETRY ou GEOGRAPHY.

Type de retour

DOUBLE PRECISION

Si geo est null, null est renvoyé.

Exemples

Le SQL suivant renvoie la zone cartésienne d'un multipolygone.

```
SELECT ST_Area(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((10 0,20 0,20 10,10
 0)))'));
```

```
 st_area
---------
     100
```

Le code SQL suivant renvoie la zone d'un polygone point dans une géographie.

```
SELECT ST_Area(ST_GeogFromText('polygon((34 35, 28 30, 25 34, 34 35))'));
```

```
     st_area
-----------------
 201824655743.383
```

Le code SQL suivant renvoie zéro pour une géographie linéaire.

```
SELECT ST_Area(ST_GeogFromText('multipoint(0 0, 1 1, -21.32 121.2)'));
```

```
 st_area
---------
       0
```

ST_ AsBinary

ST_ AsBinary renvoie la représentation binaire hexadécimale connue (WKB) d'une géométrie d'entrée. Pour les géométries 3DZ, 3DM et 4D, ST_ AsBinary utilise la valeur standard de l'Open Geospatial Consortium (OGC) pour le type de géométrie.

Syntaxe

```
ST_AsBinary(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`VARBYTE`

Si geom est null, null est renvoyé.

Exemples

Le SQL suivant renvoie la représentation WKB hexadécimale d'un polygone.

```
SELECT ST_AsBinary(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_asbinary
--------------------------------
0103000000010000000500000000000000000000000000000000000000000000000000F03F00000
```

ST_AsEWKB

ST_AsEWKT renvoie la représentation EWKT étendue d'une géométrie en entrée. Pour les géométries 3DZ, 3DM et 4D, ST_AsEWKB utilise la valeur standard OGC (Open Geospatial Consortium) pour le type de géométrie.

Syntaxe

```
ST_AsEWKB(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

VARBYTE

Si geom est null, null est renvoyé.

Exemples

Le SQL suivant renvoie la représentation EWKB hexadécimale d'un polygone.

```
SELECT ST_AsEWKB(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_asewkb
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000F
```

ST_AsEWKT

ST_AsEWKT renvoie la représentation (EWKT) d'une géométrie ou d'une géographie d'entrée. Pour les géométries 3DZ, 3DM et 4D, ST_AsEWKT ajoute Z, M ou ZM à la valeur WKT du type de géométrie.

Syntaxe

```
ST_AsEWKT(geo)
```

```
ST_AsEWKT(geo, precision)
```

Arguments

geo

Valeur de type de données GEOMETRY ou GEOGRAPHY ou expression qui est évaluée sur un type GEOMETRY ou GEOGRAPHY.

precision

Valeur du type de données INTEGER. Pour les géométries, les coordonnées de geo sont affichées à l'aide de la précision spécifiée 1-20. Si la précision n'est pas spécifiée, la valeur est 15. Pour les géométries, les coordonnées de geo sont affichées à l'aide de la précision spécifiée. Si la précision n'est pas spécifiée, la valeur est 15.

Type de retour

VARCHAR

Si geo est null, null est renvoyé.

Si precision est null, null est renvoyé.

Si le résultat est supérieur à 64 Ko VARCHAR, une erreur est renvoyée.

Exemples

Le SQL suivant renvoie la représentation EWKT d'une linestring.

```
SELECT ST_AsEWKT(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326));
```

```
st_asewkt
-------------------------------
 SRID=4326;LINESTRING(3.14159265358979 -6.28318530717959,2.71828182845905
 -1.41421356237309)
```

Le SQL suivant renvoie la représentation EWKT d'une linestring. Les coordonnées des géométries sont affichées avec six chiffres de précision.

```
SELECT ST_AsEWKT(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326), 6);
```

```
st_asewkt
-------------------------------
 SRID=4326;LINESTRING(3.14159 -6.28319,2.71828 -1.41421)
```

Le code SQL suivant renvoie la représentation EWKT d'une géographie.

```
SELECT ST_AsEWKT(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
                 st_asewkt
-----------------------------------------------
 SRID=4326;LINESTRING(110 40,2 3,-10 80,-7 9)
```

ST_JSON AsGeo

ST_ AsGeo JSON renvoie la représentation GeoJSON d'une géométrie ou d'une géographie en entrée. Pour plus d'informations sur GeoJSON, consultez GeoJSON dans Wikipedia.

Pour les géométries 3DZ et 4D, la géométrie en sortie est une projection 3DZ de la géométrie 3DZ ou 4D en entrée. Autrement dit, les coordonnées x, y et z sont présentes dans la sortie. Pour les

géométries 3DM, la géométrie en sortie est une projection 2D de la géométrie 3DM en entrée. Cela signifie que seules les coordonnées x et y sont présentes dans la sortie.

Pour les géographies en entrée, ST_ AsGeo JSON renvoie la représentation GeoJSON d'une géographie en entrée. Les coordonnées de la géographie sont affichées à l'aide de la précision spécifiée.

Syntaxe

```
ST_AsGeoJSON(geo)
```

```
ST_AsGeoJSON(geo, precision)
```

Arguments

geo

Valeur de type de données GEOMETRY ou GEOGRAPHY ou expression qui est évaluée sur un type GEOMETRY ou GEOGRAPHY.

precision

Valeur du type de données INTEGER. Pour les géométries, les coordonnées de geo sont affichées à l'aide de la précision spécifiée 1-20. Si la précision n'est pas spécifiée, la valeur est 15. Pour les géométries, les coordonnées de geo sont affichées à l'aide de la précision spécifiée. Si la précision n'est pas spécifiée, la valeur est 15.

Type de retour

VARCHAR

Si geo est null, null est renvoyé.

Si precision est null, null est renvoyé.

Si le résultat est supérieur à 64 Ko VARCHAR, une erreur est renvoyée.

Exemples

Le SQL suivant renvoie la représentation GeoJSON d'une linestring.

```
SELECT ST_AsGeoJSON(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)'));
```

```
st_asgeojson
-----------------------------------------------------------------------------------------------------
 {"type":"LineString","coordinates":[[3.14159265358979,-6.28318530717959],
[2.71828182845905,-1.414213562373095]]}
```

Le SQL suivant renvoie la représentation GeoJSON d'une linestring. Les coordonnées des géométries sont affichées avec six chiffres de précision.

```
SELECT ST_AsGeoJSON(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)'), 6);
```

```
st_asgeojson
-------------------------------------------------------------------------
  {"type":"LineString","coordinates":[[3.14159,-6.28319],[2.71828,-1.41421]]}
```

Le code SQL suivant renvoie la représentation GeoJSON d'une géographie.

```
SELECT ST_AsGeoJSON(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
                            st_asgeojson
-------------------------------------------------------------------
 {"type":"LineString","coordinates":[[110,40],[2,3],[-10,80],[-7,9]]}
```

ST_WKB AsHex

ST_ AsHex WKB renvoie la représentation binaire hexadécimale connue (WKB) d'une géométrie ou d'une géographie en entrée à l'aide de caractères hexadécimaux ASCII (0—9, A—F). Pour les géométries ou les géographies 3DZ, 3DM et 4D, ST_ AsHex WKB utilise la valeur standard de l'Open Geospatial Consortium (OGC) pour le type de géométrie ou de géographie.

Syntaxe

```
ST_AsHexWKB(geo)
```

Arguments

geo

Valeur de type de données GEOMETRY ou GEOGRAPHY ou expression qui est évaluée sur un type GEOMETRY ou GEOGRAPHY.

Type de retour

VARCHAR

Si geo est null, null est renvoyé.

Si le résultat est supérieur à 64 Ko VARCHAR, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie la représentation WKB hexadécimale d'un polygone dans une géométrie.

```
SELECT ST_AsHexWKB(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_ashexwkb
--------------------------------
0103000000010000000500000000000000000000000000000000000000000000000000F03F00000
```

Le code SQL suivant renvoie la représentation WKB hexadécimale d'un polygone dans une géographie.

```
SELECT ST_AsHexWKB(ST_GeogFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))'));
```

```
st_ashexwkb
--------------------------------
0103000000010000000500000000000000000000000000000000000000000000000000F03F00000
```

ST_WEBB AsHex

ST_ AsHex EWKB renvoie la représentation binaire connue étendue (EWKB) d'une géométrie ou d'une géographie en entrée à l'aide de caractères hexadécimaux ASCII (0—9, A—F). Pour les

géométries ou les géographies 3DZ, 3DM et 4D, ST_ EWKB AsHex utilise la valeur WKB étendue de PostGIS pour le type de géométrie ou de géographie.

Syntaxe

```
ST_AsHexEWKB(geo)
```

Arguments

geo

Valeur de type de données GEOMETRY ou GEOGRAPHY ou expression qui est évaluée sur un type GEOMETRY ou GEOGRAPHY.

Type de retour

VARCHAR

Si geo est null, null est renvoyé.

Si le résultat est supérieur à 64 Ko VARCHAR, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie la représentation EWKB hexadécimale d'un polygone dans une géométrie.

```
SELECT ST_AsHexEWKB(ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326));
```

```
st_ashexewkb
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000F
```

Le code SQL suivant renvoie la représentation EWKB hexadécimale d'un polygone dans une géographie.

```
SELECT ST_AsHexEWKB(ST_GeogFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))'));
```

```
st_ashexewkb
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000000000000000000000000000000F
```

ST_ AsText

ST_ AsText renvoie la représentation textuelle connue (WKT) d'une géométrie ou d'une géographie en entrée. Pour les géométries ou les géographies 3DZ, 3DM et 4D, ST_AsEWKT ajoute Z, M ou ZM à la valeur WKT du type de géométrie ou de géographie.

Syntaxe

```
ST_AsText(geo)
```

```
ST_AsText(geo, precision)
```

Arguments

geo

Valeur de type de données `GEOMETRY` ou `GEOGRAPHY` ou expression qui est évaluée sur un type `GEOMETRY` ou `GEOGRAPHY`.

precision

Valeur du type de données `INTEGER`. Pour les géométries, les coordonnées de geo sont affichées à l'aide de la précision spécifiée 1-20. Si la précision n'est pas spécifiée, la valeur est 15. Pour les géographies, les coordonnées de geo sont affichées à l'aide de la précision spécifiée. Si la précision n'est pas spécifiée, la valeur est 15.

Type de retour

`VARCHAR`

Si geo est null, null est renvoyé.

Si precision est null, null est renvoyé.

Si le résultat est supérieur à 64 Ko `VARCHAR`, une erreur est renvoyée.

Exemples

Le SQL suivant renvoie la représentation WKT d'une linestring.

```
SELECT ST_AsText(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326));
```

```
st_astext
-------------------------------
LINESTRING(3.14159265358979 -6.28318530717959,2.71828182845905 -1.41421356237309)
```

Le SQL suivant renvoie la représentation WKT d'une linestring. Les coordonnées des géométries sont affichées avec six chiffres de précision.

```
SELECT ST_AsText(ST_GeomFromText('LINESTRING(3.141592653589793
 -6.283185307179586,2.718281828459045 -1.414213562373095)', 4326), 6);
```

```
st_astext
-----------------------------------------------
 LINESTRING(3.14159 -6.28319,2.71828 -1.41421)
```

Le code SQL suivant renvoie la représentation WKT d'une géographie.

```
SELECT ST_AsText(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
             st_astext
-----------------------------------
 LINESTRING(110 40,2 3,-10 80,-7 9)
```

ST_Azimuth

ST_Azimuth renvoie l'azimuth cartésien basé au nord à l'aide des projections 2D des deux points d'entrée.

Syntaxe

```
ST_Azimuth(point1, point2)
```

Arguments

point1

Valeur POINT du type de données GEOMETRY. L'identifiant système de référence spatiale (SRID) du point1 doit correspondre au SRID du point2.

point2

Valeur POINT du type de données GEOMETRY. Le SRID du point2 doit correspondre au SRID du point1.

Type de retour

Nombre qui est un angle en radians du type de données DOUBLE PRECISION. Les valeurs vont de 0 (inclus) à 2 pi (exclus).

Si point1 ou point2 n'est pas un point vide, une erreur est renvoyée.

Si point1 ou point2 est null, null est renvoyé.

Si point1 et point2 sont égaux, null est renvoyé.

Si point1 ou point2 n'est pas un point, une erreur est renvoyée.

Si point1 et point2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Exemples

Le SQL suivant renvoie l'azimuth des points d'entrée.

```
SELECT ST_Azimuth(ST_Point(1,2), ST_Point(5,6));
```

```
st_azimuth
-------------------
 0.7853981633974483
```

ST_Boundary

ST_Boundary renvoie la limite d'une géométrie d'entrée comme suit :

- Si la géométrie en entrée est vide (c'est-à-dire qu'elle ne contient aucun point), elle est renvoyée telle quelle.
- Si la géométrie en entrée est un point ou un multipoint non vide, une collection de géométries vide est renvoyée.
- Si l'entrée est une linestring ou une multilinestring, un multipoint contenant tous les points de la limite est renvoyé. Le multipoint peut être vide.
- Si l'entrée est un polygone qui n'a pas d'anneaux intérieurs, une linestring fermée représentant sa limite est renvoyée.
- Si l'entrée est un polygone qui a des anneaux intérieurs ou un multipolygone, une multilinestring est renvoyée. La valeur multilinestring contient toutes les limites de tous les anneaux de la géométrie de surface sous forme de linestrings fermées.

Pour déterminer l'égalité des points, ST_Boundary opère sur la projection 2D de la géométrie en entrée. Si la géométrie en entrée est vide, une copie de celle-ci est renvoyée dans la même dimension que l'entrée. Pour les géométries 3DM et 4D non vides, leurs coordonnées `m` sont supprimées. Dans le cas particulier des multilinestrings 3DZ et 4D, les coordonnées `z` des points limites de la multilinestring sont calculés comme les moyennes des valeurs z distinctes des points limites linéaires avec la même projection 2D.

Syntaxe

```
ST_Boundary(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`GEOMETRY`

Si geom est null, null est renvoyé.

Si geom n'est pas un `GEOMETRYCOLLECTION`, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie la limite du polygone en entrée en tant que multilinestring.

```
SELECT ST_AsEWKT(ST_Boundary(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(1 1,1
 2,2 1,1 1))')));
```

```
st_asewkt
--------------------
 MULTILINESTRING((0 0,10 0,10 10,0 10,0 0),(1 1,1 2,2 1,1 1))
```

ST_Buffer

ST_Buffer renvoie une géométrie 2D qui représente tous les points dont la distance par rapport à la géométrie d'entrée projetée sur le plan cartésien XY est inférieure ou égale à la distance d'entrée.

Syntaxe

```
ST_Buffer(geom, distance)
```

```
ST_Buffer(geom, distance, number_of_segments_per_quarter_circle)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

distance

Valeur du type de données `DOUBLE PRECISION` qui représente la distance (ou le rayon) du tampon.

number_of_segments_per_quarter_circle

Valeur du type de données `INTEGER`. Cette valeur détermine le nombre de points à approcher d'un quart de cercle autour de chaque sommet de la géométrie d'entrée. Les valeurs négatives ont pour valeur par défaut zéro. La valeur par défaut est de 8.

Type de retour

GEOMETRY

La fonction ST_Buffer renvoie une géométrie bidimensionnelle (2D) dans le plan cartésien XY.

Si geom n'est pas un GEOMETRYCOLLECTION, une erreur est renvoyée.

Exemples

Le SQL suivant renvoie le sous-type de la géométrie de linestring d'entrée.

```
SELECT ST_AsEwkt(ST_Buffer(ST_GeomFromText('LINESTRING(1 2,5 2,5 8)'), 2));
```

```
               st_asewkt
POLYGON((-1 2,-0.96157056080646 2.39018064403226,-0.847759065022573
 2.76536686473018,-0.662939224605089 3.11114046603921,-0.414213562373093
 3.4142135623731,-0.111140466039201 3.66293922460509,0.234633135269824
 3.84775906502257,0.609819355967748 3.96157056080646,1 4,3 4,3 8,3.03842943919354
 8.39018064403226,3.15224093497743 8.76536686473018,3.33706077539491
 9.11114046603921,3.58578643762691 9.4142135623731,3.8888595339608
 9.66293922460509,4.23463313526982 9.84775906502257,4.60981935596775
 9.96157056080646,5 10,5.39018064403226 9.96157056080646,5.76536686473018
 9.84775906502257,6.11114046603921 9.66293922460509,6.4142135623731
 9.41421356237309,6.66293922460509 9.1111404660392,6.84775906502258
 8.76536686473017,6.96157056080646 8.39018064403225,7 8,7 2,6.96157056080646
 1.60981935596774,6.84775906502257 1.23463313526982,6.66293922460509
 0.888859533960796,6.41421356237309 0.585786437626905,6.1111404660392
 0.33706077539491,5.76536686473018 0.152240934977427,5.39018064403226
 0.0384294391935391,5 0,1 0,0.609819355967744 0.0384294391935391,0.234633135269821
 0.152240934977427,-0.111140466039204 0.337060775394909,-0.414213562373095
 0.585786437626905,-0.662939224605091 0.888859533960796,-0.847759065022574
 1.23463313526982,-0.961570560806461 1.60981935596774,-1 2))
```

Le SQL suivant renvoie le tampon de la géométrie du point d'entrée qui se rapproche d'un cercle. Étant donné que la commande ne spécifie pas le nombre de segments par quart de cercle, la fonction utilise la valeur par défaut de huit segments pour approcher le quart de cercle.

```
SELECT ST_AsEwkt(ST_Buffer(ST_GeomFromText('POINT(3 4)'), 2));
```

```
               st_asewkt
```

```
POLYGON((1 4,1.03842943919354 4.39018064403226,1.15224093497743
 4.76536686473018,1.33706077539491 5.11114046603921,1.58578643762691
 5.4142135623731,1.8888595339608 5.66293922460509,2.23463313526982
 5.84775906502257,2.60981935596775 5.96157056080646,3 6,3.39018064403226
 5.96157056080646,3.76536686473019 5.84775906502257,4.11114046603921
 5.66293922460509,4.4142135623731 5.41421356237309,4.66293922460509
 5.1111404660392,4.84775906502258 4.76536686473017,4.96157056080646 4.39018064403225,5
 4,4.96157056080646 3.60981935596774,4.84775906502257 3.23463313526982,4.66293922460509
 2.8888595339608,4.41421356237309 2.58578643762691,4.1111404660392
 2.33706077539491,3.76536686473018 2.15224093497743,3.39018064403226 2.03842943919354,3
 2,2.60981935596774 2.03842943919354,2.23463313526982 2.15224093497743,1.8888595339608
 2.33706077539491,1.58578643762691 2.58578643762691,1.33706077539491
 2.8888595339608,1.15224093497743 3.23463313526982,1.03842943919354 3.60981935596774,1
 4))
```

Le SQL suivant renvoie le tampon de la géométrie du point d'entrée qui se rapproche d'un cercle. Étant donné que la commande spécifie 3 comme nombre de segments par quart de cercle, la fonction utilise trois segments pour approcher le quart de cercle.

```
SELECT ST_AsEwkt(ST_Buffer(ST_GeomFromText('POINT(3 4)'), 2, 3));
```

```
               st_asewkt
POLYGON((1 4,1.26794919243112 5,2 5.73205080756888,3 6,4
 5.73205080756888,4.73205080756888 5,5 4,4.73205080756888 3,4 2.26794919243112,3 2,2
 2.26794919243112,1.26794919243112 3,1 4))
```

ST_Centroid

ST_CentroID renvoie un point qui représente un centroïde d'une géométrie comme suit :

- Pour les géométries `POINT`, il renvoie le point dont les coordonnées sont la moyenne des coordonnées des points de la géométrie.
- Pour les géométries `LINESTRING`, il renvoie le point dont les coordonnées sont la moyenne pondérée des points médians des segments de la géométrie, où les poids sont les longueurs des segments de la géométrie.
- Pour les géométries `POLYGON`, il renvoie le point dont les coordonnées sont la moyenne pondérée des centroïdes d'une triangulation de la géométrie surfacique où les poids sont les zones des triangles dans la triangulation.

- Pour les collections de géométrie, elle renvoie la moyenne pondérée des centroïdes des géométries de dimension topologique maximale de la collection de géométries.

Syntaxe

```
ST_Centroid(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

GEOMETRY

Si geom est null, null est renvoyé.

Si geom est vide, null est renvoyé.

Exemples

Le code SQL suivant renvoie un point central d'une linestring d'entrée.

```
SELECT ST_AsEWKT(ST_Centroid(ST_GeomFromText('LINESTRING(110 40, 2 3, -10 80, -7 9, -22
 -33)', 4326)))
```

```
                    st_asewkt
-----------------------------------------------------
 SRID=4326;POINT(15.6965103455214 27.0206782881905)
```

ST_Collect

ST_Collect a deux variantes. L'une accepte deux géométries et l'autre accepte une expression agrégée.

La première variante de ST_Collect crée une géométrie à partir des géométries en entrée. L'ordre des géométries en entrée est conservé. Cette variante fonctionne comme suit :

- Si les deux géométries en entrée sont des valeurs point, alors une valeur MULTIPOINT avec deux valeurs point est renvoyée.
- Si les deux géométries en entrée sont des valeurs linestring, alors une valeur MULTILINESTRING avec deux valeurs linestrings est renvoyée.
- Si les deux géométries en entrée sont des polygones, alors un MULTIPOLYGON avec deux polygones est renvoyé.
- Sinon, une valeur GEOMETRYCOLLECTION avec deux géométries en entrée est renvoyée.

La deuxième variante de ST_Collect crée une géométrie à partir de géométries dans une colonne de géométrie. Il n'y a pas d'ordre de renvoi déterminé des géométries. Spécifiez la clause WITHIN GROUP (ORDER BY...) pour spécifier l'ordre des géométries renvoyées. Cette variante fonctionne comme suit :

- Si toutes les lignes non NULL de l'expression d'agrégation en entrée sont des points, un multipoint contenant tous les points de l'expression d'agrégation est renvoyé.
- Si toutes les lignes non NULL de l'expression agrégée sont des linestrings, une multilinestring contenant toutes les linestrings de l'expression agrégée est renvoyée.
- Si toutes les lignes non NULL de l'expression agrégée sont des polygones, un multipolygone contenant tous les polygones de l'expression agrégée est renvoyé.
- Sinon, une GEOMETRYCOLLECTION contenant toutes les géométries de l'expression agrégée est renvoyée.

ST_Collect renvoie la géométrie de la même dimension que les géométries en entrée. Toutes les géométries en entrée doivent être de la même dimension.

Syntaxe

```
ST_Collect(geom1, geom2)
```

```
ST_Collect(aggregate_expression)  [WITHIN GROUP (ORDER BY sort_expression1 [ASC | DESC]
 [, sort_expression2 [ASC | DESC] ...])]
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

aggregate_expression

Colonne de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

[WITHIN GROUP (ORDER BY sort_expression1 [ASC | DESC] [, sort_expression2 [ASC | DESC] ...])]

Clause facultative qui spécifie l'ordre de tri des valeurs regroupées. La clause ORDER BY contient une liste d'expressions de tri. Les expressions de tri sont des expressions similaires aux expressions de tri valides dans une liste de sélection de requête, telles qu'un nom de colonne. Vous pouvez spécifier un ordre de tri croissant (ASC) ou décroissant (DESC). L'argument par défaut est ASC.

Type de retour

GEOMETRY de sous-type MULTIPOINT, MULTILINESTRING, MULTIPOLYGON ou GEOMETRYCOLLECTION.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID des géométries d'entrée.

Si geom1 ou geom2 est null, null est renvoyé.

Si toutes les lignes de aggregate_expression sont null, null est renvoyé.

Si geom1 est null, une copie de geom2 est renvoyée. De même, si geom2 est null, une copie de geom1 est renvoyée.

Si geom1 et geom2 ont des valeurs SRID différentes, une erreur est renvoyée.

Si deux géométries dans aggregate_expression ont des valeurs SRID différentes, une erreur est renvoyée.

Si la géométrie renvoyée est supérieure à la taille maximale d'une GEOMETRY, une erreur est renvoyée.

Si geom1 et geom2 ont des dimensions différentes, une erreur est renvoyée.

Si deux géométries dans aggregate_expression ont des dimensions différentes, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie une collection de géométries qui contient les deux géométries d'entrée.

```
SELECT ST_AsText(ST_Collect(ST_GeomFromText('LINESTRING(0 0,1 1)'),
 ST_GeomFromText('POLYGON((10 10,20 10,10 20,10 10))')));
```

```
st_astext
-----------
 GEOMETRYCOLLECTION(LINESTRING(0 0,1 1),POLYGON((10 10,20 10,10 20,10 10)))
```

Le code SQL suivant collecte toutes les géométries d'une table dans une collection de géométries.

```
WITH tbl(g) AS (SELECT ST_GeomFromText('POINT(1 2)', 4326) UNION ALL
SELECT ST_GeomFromText('LINESTRING(0 0,10 0)', 4326) UNION ALL
SELECT ST_GeomFromText('MULTIPOINT(13 4,8 5,4 4)', 4326) UNION ALL
SELECT NULL::geometry UNION ALL
SELECT ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))', 4326))
SELECT ST_AsEWKT(ST_Collect(g)) FROM tbl;
```

```
st_astext
-----------
 SRID=4326;GEOMETRYCOLLECTION(POINT(1 2),LINESTRING(0 0,10 0),MULTIPOINT((13 4),(8 5),
(4 4)),POLYGON((0 0,10 0,0 10,0 0)))
```

Le code SQL suivant collecte toutes les géométries de la table regroupées par la colonne id et classées par cet ID. Dans cet exemple, les géométries résultantes sont regroupées par ID comme suit :

- id 1 — points dans un multipoint.

- id 2 : linestrings dans une multilinestring.
- id 3 — sous-types mixtes d'une collection de géométries.
- id 4 — polygones dans un multipolygone.
- id 5 — null et le résultat est null.

```
WITH tbl(id, g) AS (SELECT 1, ST_GeomFromText('POINT(1 2)', 4326) UNION ALL
SELECT 1, ST_GeomFromText('POINT(4 5)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(0 0,10 0)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(10 0,20 -5)', 4326) UNION ALL
SELECT 3, ST_GeomFromText('MULTIPOINT(13 4,8 5,4 4)', 4326) UNION ALL
SELECT 3, ST_GeomFromText('MULTILINESTRING((-1 -1,-2 -2),(-3 -3,-5 -5))', 4326) UNION
 ALL
SELECT 4, ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))', 4326) UNION ALL
SELECT 4, ST_GeomFromText('POLYGON((20 20,20 30,30 20,20 20))', 4326) UNION ALL
SELECT 1, NULL::geometry UNION ALL SELECT 2, NULL::geometry UNION ALL
SELECT 5, NULL::geometry UNION ALL SELECT 5, NULL::geometry)
SELECT id, ST_AsEWKT(ST_Collect(g)) FROM tbl GROUP BY id ORDER BY id;
```

```
 id |                                                st_asewkt

----
+-----------------------------------------------------------------------------------------------
  1 | SRID=4326;MULTIPOINT((1 2),(4 5))
  2 | SRID=4326;MULTILINESTRING((0 0,10 0),(10 0,20 -5))
  3 | SRID=4326;GEOMETRYCOLLECTION(MULTIPOINT((13 4),(8 5),(4 4)),MULTILINESTRING((-1
 -1,-2 -2),(-3 -3,-5 -5)))
  4 | SRID=4326;MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((20 20,20 30,30 20,20 20)))
  5 |
```

Le SQL suivant collecte toutes les géométries d'une table d'une collection de géométries. Les résultats sont classés par ordre décroissant par `id`, puis classés de manière lexicographique en fonction de leurs coordonnées x minimales et maximales.

```
WITH tbl(id, g) AS (
SELECT 1, ST_GeomFromText('POINT(4 5)', 4326) UNION ALL
SELECT 1, ST_GeomFromText('POINT(1 2)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(10 0,20 -5)', 4326) UNION ALL
SELECT 2, ST_GeomFromText('LINESTRING(0 0,10 0)', 4326) UNION ALL
```

```
SELECT 3, ST_GeomFromText('MULTIPOINT(13 4,8 5,4 4)', 4326) UNION ALL
SELECT 3, ST_GeomFromText('MULTILINESTRING((-1 -1,-2 -2),(-3 -3,-5 -5))', 4326) UNION
 ALL
SELECT 4, ST_GeomFromText('POLYGON((20 20,20 30,30 20,20 20))', 4326) UNION ALL
SELECT 4, ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))', 4326) UNION ALL
SELECT 1, NULL::geometry UNION ALL SELECT 2, NULL::geometry UNION ALL
SELECT 5, NULL::geometry UNION ALL SELECT 5, NULL::geometry)
SELECT ST_AsEWKT(ST_Collect(g) WITHIN GROUP (ORDER BY id DESC, ST_XMin(g), ST_XMax(g)))
 FROM tbl;
```

```
                          st_asewkt

-----------------------------------------------------------------------------------------------
 SRID=4326;GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0 10,0 0)),POLYGON((20 20,20 30,30
 20,20 20)),MULTILINESTRING((-1 -1,-2 -2),(-3 -3,-5 -5)),MULTIPOINT((13 4),(8 5),(4
 4)),LINESTRING(0 0,10 0),LINESTRING(10 0,20 -5),POINT(1 2),POINT(4 5)
```

ST_Contains

ST_Contains renvoie true si la projection 2D de la première géométrie en entrée contient la projection 2D de la deuxième géométrie en entrée. La géométrie A contient la géométrie B si chaque point dans B est un point dans A, et leur intérieur comporte une intersection non vide.

ST_Contains(A, B) équivaut à ST_Within(B, A).

Syntaxe

```
ST_Contains(geom1, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Cette valeur est comparée à geom1 afin de déterminer si elle est contenue dans geom1.

Type de retour

BOOLEAN

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le SQL suivant vérifie si le premier polygone contient le deuxième polygone.

```
SELECT ST_Contains(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_contains
-----------
 false
```

ST_ ContainsProperly

ST_ ContainsProperly renvoie la valeur true si les deux géométries en entrée ne sont pas vides et que tous les points de la projection 2D de la deuxième géométrie sont des points intérieurs de la projection 2D de la première géométrie.

Syntaxe

```
ST_ContainsProperly(geom1, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Le sous-type ne peut pas être GEOMETRYCOLLECTION.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Le sous-type ne peut pas être GEOMETRYCOLLECTION. Cette valeur est comparée à geom1 pour déterminer si tous ses points sont des points intérieurs de geom1.

Type de retour

BOOLEAN

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie les valeurs de ST_Contains et ST_ ContainsProperly lorsque la chaîne de ligne d'entrée croise l'intérieur et la limite du polygone en entrée (mais pas son extérieur). Le polygone contient la linestring, mais ne contient pas correctement la linestring.

```
WITH tmp(g1, g2)
AS (SELECT ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0))'),
 ST_GeomFromText('LINESTRING(5 5,10 5,10 6,5 5)')) SELECT ST_Contains(g1, g2),
 ST_ContainsProperly(g1, g2)
FROM tmp;
```

```
 st_contains | st_containsproperly
-------------+---------------------
 t           | f
```

ST_ ConvexHull

ST_ ConvexHull renvoie une géométrie qui représente l'enveloppe convexe des points non vides contenus dans la géométrie d'entrée.

Pour une entrée vide, la géométrie résultante est identique à la géométrie en entrée. Pour toutes les entrées non vides, la fonction fonctionne sur la projection 2D de la géométrie en entrée.

Toutefois, la dimension de la géométrie en sortie dépend de la dimension de la géométrie en entrée. Plus précisément, lorsque la géométrie en entrée est une géométrie 3DM ou 3D non vide, les coordonnées `m` sont supprimées. Autrement dit, la dimension de la géométrie renvoyée est 2D ou 3DZ, respectivement. Si l'entrée est une géométrie 2D ou 3DZ non vide, la géométrie résultante a la même dimension.

Syntaxe

```
ST_ConvexHull(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`GEOMETRY`

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID de la géométrie d'entrée.

Si geom est null, null est renvoyé.

Les valeurs renvoyées sont comme suit :

Nombre de points sur l'enveloppe convexe	Sous-type de géométrie
0	Une copie de geom est renvoyée.
1	Un sous-type `POINT` est renvoyé.
2	Un sous-type `LINESTRING` est renvoyé. Les deux points de la linestring renvoyée sont classés de manière lexicographique.
3 ou plus	Un sous-type `POLYGON` sans anneaux intérieurs est renvoyé. Le polygone est orienté dans le sens des aiguilles d'une montre, et le premier

Nombre de points sur l'enveloppe convexe	Sous-type de géométrie
	point de l'anneau extérieur est le point lexicographique le plus petit de l'anneau.

Exemples

Le code SQL suivant renvoie la représentation de texte connu étendu (EWKT) d'une linestring. Dans ce cas, l'enveloppe convexe renvoyée est un polygone.

```
SELECT ST_AsEWKT(ST_ConvexHull(ST_GeomFromText('LINESTRING(0 0,1 0,0 1,1 1,0.5 0.5)')))
 as output;
```

```
output
-------------
POLYGON((0 0,0 1,1 1,1 0,0 0))
```

Le SQL suivant renvoie la représentation EWKT d'une linestring. Dans ce cas, l'enveloppe convexe renvoyée est une linestring.

```
SELECT ST_AsEWKT(ST_ConvexHull(ST_GeomFromText('LINESTRING(0 0,1 1,0.2 0.2,0.6 0.6,0.5
 0.5)'))) as output;
```

```
output
-------------
LINESTRING(0 0,1 1)
```

Le code SQL suivant renvoie la représentation EWKT d'un multipoint. Dans ce cas, l'enveloppe convexe renvoyée est un point.

```
SELECT ST_AsEWKT(ST_ConvexHull(ST_GeomFromText('MULTIPOINT(0 0,0 0,0 0)'))) as output;
```

```
output
-------------
POINT(0 0)
```

ST_ CoveredBy

ST_ CoveredBy renvoie true si la projection 2D de la première géométrie en entrée est couverte par la projection 2D de la deuxième géométrie en entrée. La géométrie `A` est couverte par la géométrie `B` si les deux sont non vides et si chaque point dans `A` est un point dans `B`.

ST_ CoveredBy (`A`,`B`) est équivalent à ST_Covers (`B`,). `A`

Syntaxe

```
ST_CoveredBy(geom1, geom2)
```

Arguments

geom1

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Cette valeur est comparée à geom2 afin de déterminer si elle est couverte par geom2.

geom2

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`BOOLEAN`

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le SQL suivant vérifie si le premier polygone est couvert par le deuxième polygone.

```
SELECT ST_CoveredBy(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_coveredby
```

```
-----------
 true
```

ST_Covers

ST_Covers renvoie true si la projection 2D de la première géométrie en entrée couvre la projection 2D de la deuxième géométrie en entrée. La géométrie A couvre la géométrie B si les deux sont non vides et si chaque point dans B est un point dans A.

ST_Covers (A,B) est équivalent à ST_ CoveredBy (B,). A

Syntaxe

```
ST_Covers(geom1, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Cette valeur est comparée à geom1 afin de déterminer si elle couvre geom1.

Type de retour

BOOLEAN

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le SQL suivant vérifie si le premier polygone couvre le deuxième polygone.

```
SELECT ST_Covers(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_covers
-----------
 false
```

ST_Crosses

ST_Crosses renvoie true si les projections 2D des deux géométries en entrée se croisent.

Syntaxe

```
ST_Crosses(geom1, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

BOOLEAN

Si geom1 ou geom2 est null, une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Exemples

Le code SQL suivant vérifie si le premier polygone croise le deuxième multipoint. Dans cet exemple, le multipoint croise à la fois l'intérieur et l'extérieur du polygone, c'est pourquoi ST_Crosses renvoie true.

```
SELECT ST_Crosses (ST_GeomFromText('polygon((0 0,10 0,10 10,0 10,0 0))'),
 ST_GeomFromText('multipoint(5 5,0 0,-1 -1)'));
```

```
st_crosses
-------------
 true
```

Le code SQL suivant vérifie si le premier polygone croise le deuxième multipoint. Dans cet exemple, le multipoint croise l'extérieur du polygone mais pas son intérieur, c'est pourquoi ST_Crosses renvoie false.

```
SELECT ST_Crosses (ST_GeomFromText('polygon((0 0,10 0,10 10,0 10,0 0))'),
 ST_GeomFromText('multipoint(0 0,-1 -1)'));
```

```
st_crosses
-------------
 false
```

ST_Dimension

ST_Dimension renvoie la dimension inhérente d'une géométrie d'entrée. La dimension inhérente est la valeur de la dimension du sous-type défini dans la géométrie.

Pour les entrées de géométrie 3DM, 3DZ et 4D, ST_Dimension renvoie le même résultat que pour les entrées de géométrie 2D.

Syntaxe

```
ST_Dimension(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`INTEGER` représentant la dimension inhérente de geom.

Si geom est null, null est renvoyé.

Les valeurs renvoyées sont comme suit :

Valeur renvoyée	Sous-type de géométrie
0	Renvoyé si geom est un sous-type `POINT` ou `MULTIPOINT` .
1	Renvoyé si geom est un sous-type `LINESTRING` ou `MULTILINESTRING` .
2	Renvoyé si geom est un sous-type `POLYGON` ou `MULTIPOLYGON` .
0	Renvoyé si geom est un sous-type `GEOMETRYCOLLECTION` vide.
Dimension la plus importantes des composants de la collection.	Renvoyé si geom est un sous-type `GEOMETRYCOLLECTION`

Exemples

Le SQL suivant convertit une représentation WKT (well-known text) d'une LINESTRING à quatre points en objet GEOMETRY et renvoie la dimension de la linestring.

```
SELECT ST_Dimension(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'));
```

```
st_dimension
-------------
1
```

ST_Disjoint

ST_Disjoint renvoie true si les projections 2D des deux géométries d'entrée n'ont aucun point commun.

Syntaxe

```
ST_Disjoint(geom1, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

BOOLEAN

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le SQL suivant vérifie si le premier polygone est séparé du deuxième polygone.

```
SELECT ST_Disjoint(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(2 2,2 5,5 5,5
 2,2 2))'), ST_Point(4, 4));
```

```
st_disjoint
-----------
 true
```

ST_Distance

Pour les géométries d'entrée, ST_Distance renvoie la distance euclidienne minimale entre les projections 2D des deux valeurs géométriques d'entrée.

Pour les géométries 3DM, 3DZ, 4D, ST_Distance renvoie la distance euclidienne entre les projections 2D de deux valeurs géométriques en entrée.

Pour les géographies d'entrée, ST_Distance renvoie la distance géodésique des deux points 2D. L'unité de distance est exprimée en mètres. Pour les zones géographiques autres que les points et les points vides, une erreur est renvoyée.

Syntaxe

```
ST_Distance(geo1, geo2)
```

Arguments

geo1

Valeur de type de données GEOMETRY ou GEOGRAPHY ou expression qui est évaluée sur un type GEOMETRY ou GEOGRAPHY. Le type de données de geo1 doit être identique à geo2.

geo2

Valeur de type de données GEOMETRY ou GEOGRAPHY ou expression qui est évaluée sur un type GEOMETRY ou GEOGRAPHY. Le type de données de geo2 doit être identique à geo1.

Type de retour

DOUBLE PRECISION dans les mêmes unités que les géométries ou les géographies d'entrée.

Si geo1 ou geo2 est null ou vide, null est renvoyé.

Si geo1 et geo2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geo1 ou geo2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le SQL suivant renvoie la distance entre deux polygones.

```
SELECT ST_Distance(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 -3,-2 -1,0 -3,-1 -3))'));
```

```
  st_distance
-----------
1.4142135623731
```

Le SQL suivant renvoie la distance (en mètres) entre la porte de Brandebourg et le bâtiment du Reichstag à Berlin à l'aide d'un type de données GEOGRAPHY.

```
SELECT ST_Distance(ST_GeogFromText('POINT(13.37761826722198 52.516411678282445)'),
 ST_GeogFromText('POINT(13.377950831464005 52.51705102546893)'));
```

```
   st_distance
------------------
 74.64129172609631
```

ST_ DistanceSphere

ST_ DistanceSphere renvoie la distance entre deux géométries ponctuelles situées sur une sphère.

Syntaxe

```
ST_DistanceSphere(geom1, geom2)
```

```
ST_DistanceSphere(geom1, geom2, radius)
```

Arguments

geom1

Valeur de point en degrés d'un type de données GEOMETRY se trouvant sur une sphère. La première coordonnée du point est la valeur de la longitude. La deuxième coordonnée du point est la valeur de la latitude. Pour les géométries 3DZ, 3DM ou 4D, seules les deux premières coordonnées sont utilisées.

geom2

Valeur de point en degrés d'un type de données GEOMETRY se trouvant sur une sphère. La première coordonnée du point est la valeur de la longitude. La deuxième coordonnée du point est la valeur de la latitude. Pour les géométries 3DZ, 3DM ou 4D, seules les deux premières coordonnées sont utilisées.

rayon

Rayon d'une sphère du type de données `DOUBLE PRECISION`. Si aucun rayon n'est fourni, la sphère utilisée par défaut est la Terre et le rayon est calculé à partir de la représentation World Geodetic System (WGS) 84 de l'ellipsoïde.

Type de retour

`DOUBLE PRECISION` dans les mêmes unités que le rayon. Si aucun rayon n'est indiqué, la distance est exprimée en mètres.

Si geom1 ou geom2 est null ou vide, null est renvoyé.

Si aucun rayon n'est fourni, le résultat est en mètres le long de la surface de la Terre.

Si le rayon est un nombre négatif, une erreur est renvoyée.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 n'est pas un point, une erreur est renvoyée.

Exemples

L'exemple SQL suivant calcule la distance en kilomètres entre deux points sur Terre.

```
SELECT ROUND(ST_DistanceSphere(ST_Point(-122, 47), ST_Point(-122.1, 47.1))/ 1000, 0);
```

```
  round
-----------
 13
```

L'exemple de SQL suivant calcule les distances en kilomètres entre trois sites d'aéroport en Allemagne : Berlin Tegel (TXL), Munich International (MUC) et Frankfurt International (FRA).

```
WITH airports_raw(code,lon,lat) AS (
(SELECT 'MUC', 11.786111, 48.353889) UNION
(SELECT 'FRA', 8.570556, 50.033333) UNION
(SELECT 'TXL', 13.287778, 52.559722)),
airports1(code,location) AS (SELECT code, ST_Point(lon, lat) FROM airports_raw),
```

```
airports2(code,location) AS (SELECT * from airports1)
SELECT (airports1.code || ' <-> ' || airports2.code) AS airports,
round(ST_DistanceSphere(airports1.location, airports2.location) / 1000, 0) AS
 distance_in_km
FROM airports1, airports2 WHERE airports1.code < airports2.code ORDER BY 1;
```

```
  airports   | distance_in_km
-------------+----------------
 FRA <-> MUC |            299
 FRA <-> TXL |            432
 MUC <-> TXL |            480
```

ST_ DWithin

ST_ DWithin renvoie la valeur true si la distance euclidienne entre les projections 2D des deux valeurs de géométrie en entrée n'est pas supérieure à une valeur seuil.

Syntaxe

```
ST_DWithin(geom1, geom2, threshold)
```

Arguments

geom1

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

geom2

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

seuil

Valeur du type de données `DOUBLE PRECISION`. Cette valeur est dans les unités des arguments d’entrée.

Type de retour

`BOOLEAN`

Si geom1 ou geom2 est null, null est renvoyé.

Si le seuil est négatif, une erreur est renvoyée.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le SQL suivant vérifie si la distance entre deux polygones est comprise dans cinq unités.

```
SELECT ST_DWithin(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'),5);
```

```
st_dwithin
-----------
 true
```

ST_ EndPoint

ST_ EndPoint renvoie le dernier point d'une chaîne de ligne d'entrée. La valeur SRID (Spatial Reference System Identifier) du résultat est identique à celle de la géométrie en entrée. La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Syntaxe

```
ST_EndPoint(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type doit être `LINESTRING`.

Type de retour

`GEOMETRY`

Si geom est null, null est renvoyé.

Si geom est vide, null est renvoyé.

Si geom n'est pas un LINESTRING, null est renvoyé.

Exemples

Le SQL suivant renvoie une représentation de texte connu étendu (EWKT) d'un LINESTRING à quatre points à un objet GEOMETRY, et renvoie le point de fin de la linestring.

```
SELECT ST_AsEWKT(ST_EndPoint(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0
 5)',4326)));
```

```
st_asewkt
-------------
 SRID=4326;POINT(0 5)
```

ST_Enveloppe

ST_Envelope renvoie le cadre de délimitation minimal de la géométrie en entrée, comme suit :

- Si la géométrie en entrée est vide, la géométrie renvoyée est une copie de la géométrie en entrée.
- Si le cadre de délimitation minimal de la géométrie en entrée dégénère en un point, la géométrie renvoyée est un point.
- Si le cadre de délimitation minimal de la géométrie en entrée est unidimensionnel, une linéarité à deux points est renvoyée.
- Si aucun des éléments précédents n'est vrai, la fonction renvoie un polygone orienté dans le sens horaire dont les sommets sont les coins de la zone de délimitation minimale.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID des géométries d'entrée.

Pour toutes les entrées non vides, la fonction fonctionne sur la projection 2D de la géométrie en entrée.

Syntaxe

```
ST_Envelope(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

GEOMETRY

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant convertit une représentation de texte connu (WKT) d'un LINESTRING à quatre points en un objet GEOMETRY, et renvoie un polygone dont les sommets sont les coins de la zone de délimitation minimale.

```
SELECT ST_AsText(ST_Envelope(ST_GeomFromText('GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0
 10,0 0)),LINESTRING(20 10,20 0,10 0))')));
```

```
    st_astext
-----------------------------------
  POLYGON((0 0,0 10,20 10,20 0,0 0))
```

ST_Equals

ST_Equals renvoie true si les projections 2D des géométries d'entrée sont géométriquement égales. Les géométries sont considérées comme géométriquement égales si elles ont des ensembles de points égaux et si leur intérieur a une intersection non vide.

Syntaxe

```
ST_Equals(geom1, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Cette valeur est comparée à geom1 afin de déterminer si elle est égale à geom1.

Type de retour

BOOLEAN

Si geom1 ou geom2 est null, une erreur est renvoyée.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le SQL suivant vérifie si les deux polygones sont géométriquement égaux.

```
SELECT ST_Equals(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_equals
-----------
 false
```

Le SQL suivant vérifie si les deux linestrings sont géométriquement égales.

```
SELECT ST_Equals(ST_GeomFromText('LINESTRING(1 0,10 0)'), ST_GeomFromText('LINESTRING(1
 0,5 0,10 0)'));
```

```
st_equals
-----------
 true
```

ST_ ExteriorRing

ST_ ExteriorRing renvoie une chaîne de ligne fermée qui représente l'anneau extérieur d'un polygone en entrée. La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Syntaxe

```
ST_ExteriorRing(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

GEOMETRY du sous-type LINESTRING.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID de la géométrie d'entrée.

Si geom est null, null est renvoyé.

Si geom n'est pas un polygone, null est renvoyé.

Si geom est vide, un polygone vide est renvoyé.

Exemples

Le SQL suivant renvoie l'anneau extérieur d'un polygone en tant que linestring fermée.

```
SELECT ST_AsEWKT(ST_ExteriorRing(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17
 7,17 10,18 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12
 14,15 14,13 11,12 14))')));
```

```
st_asewkt
-----------
 LINESTRING(7 9,8 7,11 6,15 8,16 6,17 7,17 10,18 12,17 14,15 15,11 15,10 13,9 12,7 9)
```

ST_Force2D

ST_Force2D renvoie une géométrie 2D de la géométrie en entrée. Pour les géométries 2D, une copie de l'entrée est renvoyée. Pour les géométries 3DZ, 3DM et 4D, ST_Force2D projette la géométrie sur le plan XY cartésien. Les points vides de la géométrie en entrée restent des points vides dans la géométrie en sortie.

Syntaxe

```
ST_Force2D(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`GEOMETRY`.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID de la géométrie d'entrée.

Si geom est null, null est renvoyé.

Si geom est vide, une géométrie vide est renvoyée.

Exemples

Le code SQL suivant renvoie une géométrie 2D à partir d'une géométrie 3DZ.

```
SELECT ST_AsEWKT(ST_Force2D(ST_GeomFromText('MULTIPOINT Z(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT((0 1),EMPTY,(2 3),(5 6))
```

ST_Force3D

ST_Force3D est un alias pour ST_Force3DZ. Pour plus d'informations, consultez ST_Force3DZ.

ST_Force3DM

ST_Force3DM renvoie une géométrie 3DM de la géométrie en entrée. Pour les géométries 2D, les coordonnées `m` des points non vides de la géométrie en sortie sont toutes définies sur `0`. Pour les géométries 3DM, une copie de la géométrie en entrée est renvoyée. Pour les géométries 3DZ, la géométrie est projetée sur le plan XY cartésien, et les coordonnées `m` des points non vides de la géométrie en sortie sont toutes définies sur `0`. Pour les géométries 4D, la géométrie est projetée dans l'espace XYM cartésien. Les points vides de la géométrie en entrée restent des points vides dans la géométrie en sortie.

Syntaxe

```
ST_Force3DM(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`GEOMETRY`.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID de la géométrie d'entrée.

Si geom est null, null est renvoyé.

Si geom est vide, une géométrie vide est renvoyée.

Exemples

Le code SQL suivant renvoie une géométrie 3DM à partir d'une géométrie 3DZ.

```
SELECT ST_AsEWKT(ST_Force3DM(ST_GeomFromText('MULTIPOINT Z(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT M ((0 1 0),EMPTY,(2 3 0),(5 6 0))
```

ST_Force3DZ

ST_Force3DZ renvoie une géométrie 3DZ à partir de la géométrie en entrée. Pour les géométries 2D, les coordonnées z des points non vides de la géométrie en sortie sont toutes définies sur `0`. Pour les géométries 3DM, la géométrie est projetée sur le plan XY cartésien, et les coordonnées z des points non vides de la géométrie en sortie sont toutes définies sur `0`. Pour les géométries 3DZ, une copie de la géométrie en entrée est renvoyée. Pour les géométries 4D, la géométrie est projetée dans l'espace XYZ cartésien. Les points vides de la géométrie en entrée restent des points vides dans la géométrie en sortie.

Syntaxe

```
ST_Force3DZ(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`GEOMETRY`.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID de la géométrie d'entrée.

Si geom est null, null est renvoyé.

Si geom est vide, une géométrie vide est renvoyée.

Exemples

Le code SQL suivant renvoie une géométrie 3DZ à partir d'une géométrie 3DM.

```
SELECT ST_AsEWKT(ST_Force3DZ(ST_GeomFromText('MULTIPOINT M(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT Z ((0 1 0),EMPTY,(2 3 0),(5 6 0))
```

ST_Force4D

ST_Force4D renvoie une géométrie 4D de la géométrie en entrée. Pour les géométries 2D, les coordonnées `z` et `m` des points non vides de la géométrie en sortie sont toutes définies sur `0`. Pour les géométries 3DM, les coordonnées `z` des points non vides de la géométrie en sortie sont toutes définies sur `0`. Pour les géométries 3DZ, les coordonnées `m` des points non vides de la géométrie en sortie sont toutes définies sur `0`. Pour les géométries 4D, une copie de la géométrie en entrée est renvoyée. Les points vides de la géométrie en entrée restent des points vides dans la géométrie en sortie.

Syntaxe

```
ST_Force4D(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`GEOMETRY`.

La valeur de l’identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID de la géométrie d’entrée.

Si geom est null, null est renvoyé.

Si geom est vide, une géométrie vide est renvoyée.

Exemples

Le code SQL suivant renvoie une géométrie 4D à partir d'une géométrie 3DM.

```
SELECT ST_AsEWKT(ST_Force4D(ST_GeomFromText('MULTIPOINT M(0 1 2, EMPTY, 2 3 4, 5 6
 7)')));
```

```
st_asewkt
-----------
  MULTIPOINT ZM ((0 1 0 2),EMPTY,(2 3 0 4),(5 6 0 7))
```

ST_ GeoHash

ST_ GeoHash renvoie la `geohash` représentation du point d'entrée avec la précision spécifiée. La valeur de précision par défaut est de 20. Pour plus d'informations sur la définition de geohash, consultez Geohash dans Wikipédia.

Syntaxe

```
ST_GeoHash(geom)
```

```
ST_GeoHash(geom, precision)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

precision

Valeur du type de données `INTEGER`. La valeur par défaut est de 20.

Type de retour

`GEOMETRY`

La fonction renvoie la représentation `geohash` du point d'entrée.

Si le point d'entrée est vide, la fonction renvoie null.

Si la géométrie d'entrée n'est pas un point, la fonction renvoie une erreur.

Exemples

Le SQL suivant renvoie la représentation geohash d'un point d'entrée.

```
SELECT ST_GeoHash(ST_GeomFromText('POINT(45 -45)'), 25) AS geohash;
```

```
          geohash
---------------------------
 m0000000000000000000gzz
```

Le SQL suivant renvoie null parce que le point d'entrée est vide.

```
SELECT ST_GeoHash(ST_GeomFromText('POINT EMPTY'), 10) IS NULL AS result;
```

```
 result
---------
 true
```

ST_ GeogFromText

ST_ GeogFromText construit un objet géographique à partir d'une représentation en texte connu (WKT) ou en texte connu étendu (EWKT) d'une géographie en entrée.

Syntaxe

```
ST_GeogFromText(wkt_string)
```

Arguments

wkt_string

Valeur d'un type de données VARCHAR qui est une représentation WKT ou EWKT d'une géographie.

Type de retour

GEOGRAPHY

Si la valeur SRID est définie sur la valeur fournie dans l'entrée. Si la valeur SRID n'est pas fournie, elle est définie sur 4326.

Si wkt_string est null, null est renvoyé.

Si wkt_string n'est pas valide, une erreur est renvoyée.

Exemples

Le code SQL suivant construit un polygone à partir d'un objet géographique avec une valeur SRID.

```
SELECT ST_AsEWKT(ST_GeogFromText('SRID=4324;POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))'));
```

```
  st_asewkt
-----------------------------------------------
 SRID=4324;POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))
```

Le code SQL suivant construit un polygone à partir d'un objet géographique. La valeur SRID est définie sur 4326.

```
SELECT ST_AsEWKT(ST_GeogFromText('POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))'));
```

```
 st_asewkt
-----------------------------------------------
 SRID=4326;POLYGON((0 0,0 1,1 1,10 10,1 0,0 0))
```

ST_WKB GeogFrom

ST_ GeogFrom WKB construit un objet de géographie à partir d'une représentation binaire hexadécimale connue (WKB) d'une géographie en entrée.

Syntaxe

```
ST_GeogFromWKB(wkb_string)
```

Arguments

wkb_string

Valeur d'un type de données VARCHAR qui est une représentation WKB hexadécimale d'une géographie.

Type de retour

GEOGRAPHY

Si la valeur SRID est fournie, elle est définie sur la valeur fournie. Si la valeur SRID n'est pas fournie, elle est définie sur 4326.

Si wkb_string est null, null est renvoyé.

Si wkb_string n'est pas valide, une erreur est renvoyée.

Exemples

Le code SQL suivant construit une géographie à partir d'une valeur WKB hexadécimale.

```
SELECT
 ST_AsEWKT(ST_GeogFromWKB('010300000001000000050000000000000000000000000000000000000000000000000
```

```
 st_asewkt
-------------------------------------------
 SRID=4326;POLYGON((0 0,0 1,1 1,1 0,0 0))
```

ST_GeometryN

ST_GeometryN renvoie une géométrie pointée par l'index en entrée de la géométrie en entrée, comme suit :

- Si l'entrée est un point, une linestring ou un polygone, une géométrie est renvoyée telle qu'elle est si l'index est égal à un (1), et null si l'index est différent d'un (1).
- Si l'entrée est une collection multipoint, multilinestring, multipolygone ou de géométries, une collection de points, linestrings, polygones ou géométries est renvoyée telle que pointée par un index en entrée.

L'index est basé sur un. L'identificateur du système de référence spatiale (SRID) du résultat est identique à celui de la géométrie en entrée. La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Syntaxe

```
ST_GeometryN(geom, index)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

index

Valeur de type de données INTEGER qui représente la position d'un index basé sur un.

Type de retour

GEOMETRY

Si geom ou index est null, null est renvoyé.

Si l'index est hors de portée, une erreur est renvoyée.

Exemples

Le SQL suivant renvoie les géométries d'une collection de géométries.

```
WITH tmp1(idx) AS (SELECT 1 UNION SELECT 2),
tmp2(g) AS (SELECT ST_GeomFromText('GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0 10,0
 0)),LINESTRING(20 10,20 0,10 0))'))
SELECT idx, ST_AsEWKT(ST_GeometryN(g, idx)) FROM tmp1, tmp2 ORDER BY idx;
```

```
 idx |          st_asewkt
-----+-----------------------------
   1 | POLYGON((0 0,10 0,0 10,0 0))
   2 | LINESTRING(20 10,20 0,10 0)
```

ST_ GeometryType

ST_ GeometryType renvoie le sous-type d'une géométrie d'entrée sous forme de chaîne.

Pour les entrées de géométrie 3DM, 3DZ et 4D, ST_ GeometryType renvoie le même résultat que pour les entrées de géométrie 2D.

Syntaxe

```
ST_GeometryType(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

VARCHAR représentant le sous-type de geom.

Si geom est null, null est renvoyé.

Les valeurs renvoyées sont comme suit :

Valeur de chaîne renvoyée	Sous-type de géométrie
ST_Point	Renvoyé si geom est un sous-type POINT
ST_LineString	Renvoyé si geom est un sous-type LINESTRING
ST_Polygon	Renvoyé si geom est un sous-type POLYGON
ST_MultiPoint	Renvoyé si geom est un sous-type MULTIPOINT
ST_MultiLineString	Renvoyé si geom est un sous-type MULTILINESTRING

Valeur de chaîne renvoyée	Sous-type de géométrie
`ST_MultiPolygon`	Renvoyé si geom est un sous-type `MULTIPOLYGON`
`ST_GeometryCollection`	Renvoyé si geom est un sous-type `GEOMETRYCOLLECTION`

Exemples

Le SQL suivant renvoie le sous-type de la géométrie de linestring d'entrée.

```
SELECT ST_GeometryType(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'));
```

```
st_geometrytype
-------------
 ST_LineString
```

ST_WEBB GeomFrom

ST_ GeomFrom EWKB construit un objet de géométrie à partir de la représentation binaire étendue connue (EWKB) d'une géométrie d'entrée.

ST_ GeomFrom EWKB accepte les géométries 3DZ, 3DM et 4D écrites au format hexadécimal WKB et EWKB.

Syntaxe

```
ST_GeomFromEWKB(ewkb_string)
```

Arguments

ewkb_string

Valeur d'un type de données `VARCHAR` qui est une représentation EWKB hexadécimale d'une géométrie.

Type de retour

GEOMETRY

Si ewkb_string est null, null est renvoyé.

Si ewkb_string n'est pas valide, une erreur est renvoyée.

Exemples

Le SQL suivant construit un polygone à partir d'une valeur EWKB et renvoie la représentation EWKT d'un polygone.

```
SELECT
 ST_AsEWKT(ST_GeomFromEWKB('0103000020E6100000010000000500000000000000000000000000000000000000000
```

```
 st_asewkt
-------------------------------
 SRID=4326;POLYGON((0 0,0 1,1 1,1 0,0 0))
```

ST_EWKT GeomFrom

ST_ GeomFrom EWKT construit un objet géométrique à partir de la représentation étendue en texte connu (EWKT) d'une géométrie d'entrée.

ST_ GeomFrom EWKT accepte les formats 3DZ, 3DM et 4D où le type de géométrie est préfixé par Z, M ou ZM, respectivement.

Syntaxe

```
ST_GeomFromEWKT(ewkt_string)
```

Arguments

ewkt_string

Valeur du type de données VARCHAR ou expression qui correspond à un VARCHAR, c'est-à-dire une représentation EWKT d'une géométrie.

Vous pouvez utiliser le mot-clé WKT `EMPTY` pour désigner un point vide, un multipoint avec un point vide ou une collection de géométries avec un point vide. Dans l'exemple suivant, un point vide est créé.

```
ST_GeomFromEWKT('SRID=4326;POINT EMPTY');
```

Type de retour

`GEOMETRY`

Si ewkt_string est null, null est renvoyé.

Si ewkt_string n'est pas valide, une erreur est renvoyée.

Exemples

Le code SQL suivant construit une multilinestring à partir d'une valeur EWKT et renvoie une géométrie. Il renvoie également le résultat ST_AsEWKT de la géométrie.

```
SELECT ST_GeomFromEWKT('SRID=4326;MULTILINESTRING((1 0,1 0),(2 0,3 0),(4 0,5 0,6 0))')
 as geom, ST_AsEWKT(geom);
```

```
                                                                  geom

                                                 |                        st_asewkt

-------------------------------------------------------------------------------------------------
+-----------------------------------------------------------------
 0105000020E6100000030000000102000000020000000000000000F03F0000000000000000000000000000F03F00
 | SRID=4326;MULTILINESTRING((1 0,1 0),(2 0,3 0),(4 0,5 0,6 0))
```

ST_ GeomFromGeoHash

ST_ GeomFromGeoHash construit un objet géométrique à partir de la représentation géohasée d'une géométrie d'entrée. ST_ GeomFromGeoHash renvoie une géométrie bidimensionnelle (2D) dont l'identifiant de référence spatiale (SRID) est égal à zéro (0). Pour plus d'informations sur le format geohash, consultez Geohash dans Wikipédia.

Syntaxe

```
ST_GeomFromGeoHash(geohash_string)
```

```
ST_GeomFromGeoHash(geohash_string, precision)
```

Arguments

geohash_string

Valeur de type de données VARCHAR ou expression qui correspond à un type VARCHAR, qui est une représentation geohash d'une géométrie.

precision

Valeur du type de données INTEGER qui représente la précision du geohash. La valeur est le nombre de caractères du geohash à utiliser à des fins de précision. Si la valeur n'est pas spécifiée, inférieure à zéro ou supérieure à la longueur de geohash_string, alors la longueur geohash_string est utilisée.

Type de retour

GEOMETRY

Si geohash_string est null, null est renvoyé.

Si geohash_string n'est pas valide, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie un polygone de haute précision.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qqj7nmxncgyy4d0dbxqz0'));
```

```
 st_asewkt
-----------------------
 POLYGON((-115.172816 36.114646,-115.172816 36.114646,-115.172816 36.114646,-115.172816
 36.114646,-115.172816 36.114646))
```

Le code SQL suivant renvoie un point de haute précision.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qqj7nmxncgyy4d0dbxqz00'));
```

```
 st_asewkt
-----------------------
 POINT(-115.172816 36.114646)
```

Le code SQL suivant renvoie un polygone de faible précision.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qq'));
```

```
 st_asewkt
-----------------------
 POLYGON((-115.3125 35.15625,-115.3125 36.5625,-113.90625 36.5625,-113.90625
 35.15625,-115.3125 35.15625))
```

Le code SQL suivant renvoie un polygone de précision 3.

```
SELECT ST_AsText(ST_GeomFromGeoHash('9qqj7nmxncgyy4d0dbxqz0', 3));
```

```
 st_asewkt
-----------------------
 POLYGON((-115.3125 35.15625,-115.3125 36.5625,-113.90625 36.5625,-113.90625
 35.15625,-115.3125 35.15625))
```

ST_JSON GeomFromGeo

ST_ GeomFromGeo JSON construit un objet de géométrie à partir de la représentation GeoJSON d'une géométrie en entrée. Pour plus d'informations sur le format GeoJSON, consultez GeoJSON dans Wikipédia.

S'il y a au moins un point avec trois coordonnées ou plus, la géométrie résultante est 3DZ, où la composante Z est nulle pour les points qui n'ont que deux coordonnées. Si tous les points du fichier

GeoJSON en entrée contiennent deux coordonnées ou sont vides, ST_ GeomFromGeo JSON renvoie une géométrie 2D. La géométrie renvoyée est toujours dotée de l'identifiant de référence spatiale (SRID) 4326.

Syntaxe

```
ST_GeomFromGeoJSON(geojson_string)
```

Arguments

geojson_string

Une valeur de type de donnéesVARCHAR, ou une expression qui correspond à un VARCHAR type, qui est une représentation GeoJSON d'une géométrie.

Type de retour

GEOMETRY

Si geojson_string est null, null est renvoyé.

Si geojson_string n'est pas valide, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie une géométrie 2D représentée dans la représentation GeoJSON en entrée.

```
SELECT ST_AsEWKT(ST_GeomFromGeoJSON('{"type":"Point","coordinates":[1,2]}'));
```

```
 st_asewkt
-----------------------
 SRID=4326;POINT(1 2)
```

Le code SQL suivant renvoie une géométrie 3DZ représentée dans la représentation GeoJSON en entrée.

```
SELECT ST_AsEWKT(ST_GeomFromGeoJSON('{"type":"LineString","coordinates":[[1,2,3],
[4,5,6],[7,8,9]]}'));
```

```
 st_asewkt
-----------------------------------------
 SRID=4326;LINESTRING Z (1 2 3,4 5 6,7 8 9)
```

Le code SQL suivant renvoie la géométrie 3DZ quand un seul point possède trois coordonnées alors que tous les autres points ont deux coordonnées dans la représentation GeoJSON en entrée.

```
SELECT ST_AsEWKT(ST_GeomFromGeoJSON('{"type":"Polygon","coordinates":[[[0, 0],[0, 1,
 8],[1, 0],[0, 0]]]}'));
```

```
 st_asewkt
-----------------------------------------------
 SRID=4326;POLYGON Z ((0 0 0,0 1 8,1 0 0,0 0 0))
```

ST_ GeomFromGeoSquare

ST_ GeomFromGeoSquare renvoie une géométrie qui couvre la zone représentée par une valeur géosquare en entrée. La géométrie renvoyée est toujours bidimensionnelle. Pour calculer une valeur de geosquare, consultez ST_ GeoSquare.

Syntaxe

```
ST_GeomFromGeoSquare(geosquare)
```

```
ST_GeomFromGeoSquare(geosquare, max_depth)
```

Arguments

geosquare

Valeur de type de données `BIGINT` ou expression ayant pour résultat un type `BIGINT` qui est une valeur de geosquare qui décrit la séquence de subdivisions effectuées sur le domaine initial pour atteindre le carré souhaité. Cette valeur est calculée par ST_ GeoSquare.

max_depth

Valeur du type de données INTEGER qui représente le nombre maximal de subdivisions de domaine effectuées sur le domaine initial. Cette valeur doit être supérieure ou égale à 1.

Type de retour

GEOMETRY

Si geosquare n'est pas valide, la fonction renvoie une erreur.

Si l'entrée max_depth n'est pas comprise dans la plage, la fonction renvoie une erreur.

Exemples

Le code SQL suivant renvoie une géométrie à partir d'une valeur de geosquare.

```
SELECT ST_AsText(ST_GeomFromGeoSquare(797852));
```

```
 st_astext
-----------------------------------------------------------------------------------------------
 POLYGON((13.359375 52.3828125,13.359375 52.734375,13.7109375 52.734375,13.7109375
 52.3828125,13.359375 52.3828125))
```

Le code SQL suivant renvoie une géométrie à partir d'une valeur de geosquare et d'une profondeur maximale de 3.

```
SELECT ST_AsText(ST_GeomFromGeoSquare(797852, 3));
```

```
 st_astext
-------------------------------------
 POLYGON((0 45,0 90,45 90,45 45,0 45))
```

Le code SQL suivant calcule d'abord la valeur de geosquare pour Seattle en spécifiant la coordonnée X comme longitude et la coordonnée Y comme latitude (-122,3, 47,6). Il renvoie ensuite le polygone du geosquare. Bien que la sortie soit une géométrie bidimensionnelle, elle peut être utilisée pour calculer des données spatiales en termes de longitude et de latitude.

```
SELECT ST_AsText(ST_GeomFromGeoSquare(ST_GeoSquare(ST_Point(-122.3, 47.6))));
```

```
 st_astext
-----------------------------------------------------------------------------------------------------------
POLYGON((-122.335167014971 47.6080129947513,-122.335167014971
 47.6080130785704,-122.335166931152 47.6080130785704,-122.335166931152
 47.6080129947513,-122.335167014971 47.6080129947513))
```

ST_ GeomFromText

ST_ GeomFromText construit un objet géométrique à partir d'une représentation textuelle connue (WKT) d'une géométrie d'entrée.

ST_ GeomFromText accepte 3DZ, 3DM et 4D où le type de géométrie est préfixé par Z, M ou ZM, respectivement.

Syntaxe

```
ST_GeomFromText(wkt_string)
```

```
ST_GeomFromText(wkt_string, srid)
```

Arguments

wkt_string

Valeur d’un type de données VARCHAR qui est une représentation WKT d’une géométrie.

Vous pouvez utiliser le mot-clé WKT EMPTY pour désigner un point vide, un multipoint avec un point vide ou une collection de géométries avec un point vide. L’exemple suivant crée un multipoint avec un point vide et un point non vide.

```
ST_GeomFromEWKT('MULTIPOINT(1 0,EMPTY)');
```

srid

Valeur du type de données `INTEGER` qui est un identifiant de référence spatiale (SRID). Si une valeur SRID est fournie, la géométrie renvoyée a cette valeur de SRID. Sinon, la valeur de SRID de la géométrie renvoyée est définie sur 0.

Type de retour

`GEOMETRY`

Si wkt_string ou srid est null, null est renvoyé.

Si srid est négatif, null est renvoyé.

Si wkt_string n'est pas valide, une erreur est renvoyée.

Si srid n'est pas valide, une erreur est renvoyée.

Exemples

Le SQL suivant construit un objet géométrique à partir de la représentation WKT et de la valeur SRID.

```
SELECT ST_GeomFromText('POLYGON((0 0,0 1,1 1,1 0,0 0))',4326);
```

```
st_geomfromtext
--------------------------------
0103000020E6100000010000000500000000000000000000000000000000000000000000000000F0
```

ST_WKB GeomFrom

ST_ GeomFrom WKB construit un objet de géométrie à partir d'une représentation binaire hexadécimale connue (WKB) d'une géométrie d'entrée.

ST_ GeomFrom WKB accepte les géométries 3DZ, 3DM et 4D écrites au format hexadécimal WKB.

Syntaxe

```
ST_GeomFromWKB(wkb_string)
```

```
ST_GeomFromWKB(wkb_string, srid)
```

Arguments

wkb_string

Valeur d'un type de données VARCHAR qui est une représentation WKB hexadécimale d'une géométrie.

srid

Valeur du type de données INTEGER qui est un identifiant de référence spatiale (SRID). Si une valeur SRID est fournie, la géométrie renvoyée a cette valeur de SRID. Sinon, la valeur de SRID de la géométrie renvoyée est définie sur 0.

Type de retour

GEOMETRY

Si wkb_string ou srid est null, null est renvoyé.

Si srid est négatif, null est renvoyé.

Si wkb_string n'est pas valide, une erreur est renvoyée.

Si srid n'est pas valide, une erreur est renvoyée.

Exemples

Le SQL suivant construit un polygone à partir d'une valeur WKB et renvoie la représentation WKT d'un polygone.

```
SELECT
 ST_AsText(ST_GeomFromWKB('010300000001000000050000000000000000000000000000000000000000000000
```

```
 st_astext
-------------------------------
 POLYGON((0 0,0 1,1 1,1 0,0 0))
```

ST_ GeoSquare

ST_ subdivise GeoSquare récursivement le domaine ([-180, 180], [-90, 90]) en régions carrées égales appelées géoscarrés d'une profondeur spécifiée. La subdivision est basée sur l'emplacement d'un point donné. L'un des geosquares contenant le point est subdivisé à chaque étape jusqu'à atteindre la profondeur maximale. La sélection de ce geosquare est stable, c'est-à-dire que le résultat de la fonction dépend uniquement des arguments saisis. La fonction renvoie une valeur unique qui identifie le geosquare final dans lequel se trouve le point.

Le ST_ GeoSquare accepte un POINT où la coordonnée x représente la longitude et la coordonnée y représente la latitude. La longitude et la latitude sont limitées à [-180, 180] et [-90, 90], respectivement. La sortie de ST_ GeoSquare peut être utilisée comme entrée de la ST_ GeomFromGeoSquare fonction.

Il y a 360° autour de l'arc de la circonférence équatoriale de la Terre qui sont divisés en deux hémisphères (est et ouest), chacun ayant 180° de lignes longitudinales (méridiens) à partir du méridien 0°. Par convention, les longitudes orientales sont des coordonnées « + » (positives) lorsqu'elles sont projetées sur l'axe des X sur un plan cartésien et les longitudes occidentales sont des coordonnées « - » (négatives) lorsqu'elles sont projetées sur l'axe des X sur un plan cartésien. Il existe des lignes latitudinales de 90° au nord et au sud de la circonférence équatoriale 0° de la Terre, chacune étant parallèle à la circonférence équatoriale 0° de la Terre. Par convention, les lignes latitudinales nord coupent l'axe Y « + » (positif) lorsqu'elles sont projetées sur un plan cartésien et les lignes latitudinales sud coupent l'axe Y « - » (négatif) lorsqu'elles sont projetées sur un plan cartésien. La grille sphérique formée par l'intersection de lignes longitudinales et de lignes latitudinales est convertie en une grille projetée sur un plan cartésien avec des coordonnées X positives et négatives standard et des coordonnées Y positives et négatives sur le plan cartésien.

Le but de ST_ GeoSquare est de marquer ou de marquer des points proches avec des valeurs de code égales. Les points situés dans le même geosquare reçoivent la même valeur de code. Un geosquare est utilisé pour coder les coordonnées géographiques (latitude et longitude) en entier. Une région plus vaste est divisée en grilles pour délimiter une zone sur une carte avec différentes résolutions. Un geosquare peut être utilisé pour l'indexation spatiale, la discrétisation spatiale, les recherches de proximité, la recherche de lieux et la création d'identifiants de lieux uniques. La fonction ST_ GeoHash suit un processus similaire consistant à diviser une région en grilles, mais son codage est différent.

Syntaxe

```
ST_GeoSquare(geom)
```

```
ST_GeoSquare(geom, max_depth)
```

Arguments

geom

Valeur POINT de type de données GEOMETRY ou expression ayant pour résultat un sous-type POINT. La coordonnée X (longitude) du point doit se situer dans la plage suivante : -180 à 180. La coordonnée Y (latitude) du point doit se situer dans la plage suivante : -90 à 90.

max_depth

Valeur du type de données INTEGER. Nombre maximal de fois où le domaine contenant le point est subdivisé de manière récursive. La valeur doit être un entier compris entre 1 et 32. La valeur par défaut est 32. Le nombre final réel de subdivisions est inférieur ou égal à la valeur max_depth spécifiée.

Type de retour

BIGINT

La fonction renvoie une valeur unique qui identifie le geosquare final dans lequel se trouve le point d'entrée.

Si la valeur saisie pour geom n'est pas un point, la fonction renvoie une erreur.

Si la valeur saisie pour le point est vide, la valeur renvoyée n'est pas une entrée valide pour la fonction ST_ GeomFromGeoSquare. Utilisez cette ST_ IsEmpty fonction pour empêcher les appels à ST_ GeoSquare avec un point vide.

Si la valeur saisie pour le point n'est pas comprise dans la plage, la fonction renvoie une erreur.

Si la valeur saisie pour max_depth n'est pas comprise dans la plage, la fonction renvoie une erreur.

Exemples

Le code SQL suivant renvoie un geosquare depuis un point d'entrée.

```
SELECT ST_GeoSquare(ST_Point(13.5, 52.5));
```

```
  st_geosquare
-----------------------
 -4410772491521635895
```

Le code SQL suivant renvoie un geosquare depuis un point d'entrée avec une profondeur maximale de 10.

```
SELECT ST_GeoSquare(ST_Point(13.5, 52.5), 10);
```

```
 st_geosquare
--------------
 797852
```

ST_N InteriorRing

ST_ InteriorRing N renvoie une chaîne de ligne fermée correspondant à l'anneau intérieur d'un polygone d'entrée à la position d'index. La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Syntaxe

```
ST_InteriorRingN(geom, index)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

index

Valeur de type de données INTEGER qui représente la position d'un anneau d'index à basé sur un.

Type de retour

`GEOMETRY` du sous-type `LINESTRING`.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID de la géométrie d'entrée.

Si geom ou index est null, null est renvoyé.

Si l'index est hors de portée, null est renvoyé.

Si geom n'est pas un polygone, null est renvoyé.

Si geom est un polygone vide, null est renvoyé.

Exemples

Le code SQL suivant renvoie le deuxième anneau du polygone en tant que linestring fermée.

```
SELECT ST_AsEWKT(ST_InteriorRingN(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17
 7,17 10,18 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12
 14,15 14,13 11,12 14))'),2));
```

```
st_asewkt
-----------
 LINESTRING(12 14,15 14,13 11,12 14)
```

ST_Intersects

ST_Intersects renvoie true si les projections 2D des deux géométries d'entrée ont au moins un point en commun.

Syntaxe

```
ST_Intersects(geom1, geom2)
```

Arguments

geom1

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

BOOLEAN

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le SQL suivant vérifie si le premier polygone coupe le deuxième polygone.

```
SELECT ST_Intersects(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(2 2,2 5,5 5,5
 2,2 2))'), ST_GeomFromText('MULTIPOINT((4 4),(6 6))'));
```

```
st_intersects
-------------
 true
```

ST_Intersection

ST_Intersection renvoie une géométrie représentant l'intersection de deux géométries définie par des points. En d'autres termes, il renvoie la partie des deux géométries d'entrée partagées entre elles.

Syntaxe

```
ST_Intersection(geom1, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

GEOMETRY

Si geom1 et geom2 ne partagent aucun espace (ils sont disjoints), une géométrie vide est alors renvoyée.

Si geom1 ou geom2 est vide, une géométrie vide est renvoyée.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Si geom1 ou geom2 n'est pas une géométrie bidimensionnelle (2D), une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie la géométrie non vide représentant l'intersection de deux géométries d'entrée.

```
SELECT ST_AsEWKT(ST_Intersection(ST_GeomFromText('polygon((0 0,100 100,0 200,0 0))'),
 ST_GeomFromText('polygon((0 0,10 0,0 10,0 0))')));
```

```
        st_asewkt
-------------------------
 POLYGON((0 0,0 10,5 5,0 0))
```

Le code SQL suivant renvoie une géométrie vide lorsque des géométries d'entrée disjointes (non croisées) sont transmises.

```
SELECT ST_AsEWKT(ST_Intersection(ST_GeomFromText('linestring(0 100,0 0)'),
 ST_GeomFromText('polygon((1 0,10 0,1 10,1 0))')));
```

```
    st_asewkt
------------------
 LINESTRING EMPTY
```

ST_CCW IsPolygon

ST_ IsPolygon CCW renvoie true si la projection 2D du polygone ou du multipolygone en entrée est dans le sens antihoraire. Si la géométrie en entrée est un point, une linestring, un multipoint ou une multilinestring, alors la valeur true est renvoyée. Pour les collections de géométries, ST_ IsPolygon CCW renvoie la valeur true si toutes les géométries de la collection sont dans le sens antihoraire.

Syntaxe

```
ST_IsPolygonCCW(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`BOOLEAN`

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant vérifie si le polygone est dans le sens inverse des aiguilles d'une montre.

```
SELECT ST_IsPolygonCCW(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17 7,17 10,18
 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12 14,15 14,13
 11,12 14))'));
```

```
 st_ispolygonccw
----------
 true
```

ST_CW IsPolygon

ST_ IsPolygon CW renvoie true si la projection 2D du polygone ou du multipolygone en entrée est dans le sens des aiguilles d'une montre. Si la géométrie en entrée est un point, une linestring, un multipoint ou une multilinestring, alors la valeur true est renvoyée. Pour les collections de géométries, ST_ IsPolygon CW renvoie true si toutes les géométries de la collection sont dans le sens des aiguilles d'une montre.

Syntaxe

```
ST_IsPolygonCW(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`BOOLEAN`

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant vérifie si le polygone est dans le sens des aiguilles d’une montre.

```
SELECT ST_IsPolygonCW(ST_GeomFromText('POLYGON((7 9,8 7,11 6,15 8,16 6,17 7,17 10,18
 12,17 14,15 15,11 15,10 13,9 12,7 9),(9 9,10 10,11 11,11 10,10 8,9 9),(12 14,15 14,13
 11,12 14))'));
```

```
 st_ispolygonccw
----------
 true
```

ST_ IsClosed

ST_ IsClosed renvoie la valeur true si la projection 2D de la géométrie d'entrée est fermée. Les règles suivantes définissent une géométrie fermée :

- La géométrie d'entrée est un point ou un multipoint.
- La géométrie d'entrée est une linestring, et les points de début et de fin de la linestring coïncident.
- La géométrie d'entrée est une multilinestring non vide et toutes ses linestrings sont vides.
- La géométrie d'entrée est un polygone non vide, tous les anneaux du polygones sont non vides, et les points de début et de fin de tous ses anneaux coïncident.
- La géométrie d'entrée est un multipolygone non vide et tous ses polygones sont vides.
- La géométrie d'entrée est une collection géométrique non vide et tous ses composants sont fermés.

Syntaxe

```
ST_IsClosed(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`BOOLEAN`

Si geom est un point vide, false est renvoyé.

Si geom est null, null est renvoyé.

Exemples

Le SQL suivant vérifie si le polygone est fermé.

```
SELECT ST_IsClosed(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
st_isclosed
-----------
 true
```

ST_ IsCollection

ST_ IsCollection renvoie la valeur true si la géométrie d'entrée est l'un des sous-types suivants :`GEOMETRYCOLLECTION`,`MULTIPOINT`, `MULTILINESTRING` ou. `MULTIPOLYGON`

Syntaxe

```
ST_IsCollection(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`BOOLEAN`

Si geom est null, null est renvoyé.

Exemples

Le SQL suivant vérifie si le polygone est une collection.

```
SELECT ST_IsCollection(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
st_iscollection
-----------
 false
```

ST_ IsEmpty

ST_ IsEmpty renvoie true si la géométrie d'entrée est vide. Une géométrie n’est pas vide si elle contient au moins un point non vide.

ST_ IsEmpty renvoie true si la géométrie en entrée comporte au moins un point non vide.

Syntaxe

```
ST_IsEmpty(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

BOOLEAN

Si geom est null, null est renvoyé.

Exemples

Le SQL suivant vérifie si le polygone spécifié est vide.

```
SELECT ST_IsEmpty(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'));
```

```
st_isempty
-----------
 false
```

ST_ IsRing

ST_ IsRing renvoie true si la chaîne de ligne d'entrée est un anneau. Un linestring est un anneau s'il est fermé et simple.

Syntaxe

```
ST_IsRing(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. La géométrie doit être une LINESTRING.

Type de retour

BOOLEAN

Si geom n'est pas une LINESTRING, une erreur est renvoyée.

Exemples

Le code SQL suivant vérifie si la linestring spécifiée est un anneau.

```
SELECT ST_IsRing(ST_GeomFromText('linestring(0 0, 1 1, 1 2, 0 0)'));
```

```
st_isring
-----------
 true
```

ST_ IsSimple

ST_ IsSimple renvoie la valeur true si la projection 2D de la géométrie d'entrée est simple. Pour plus d'informations sur la définition d'une géométrie simple, consultez Simplicité géométrique.

Syntaxe

```
ST_IsSimple(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

BOOLEAN

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant vérifie si la linestring spécifiée est simple. Dans cet exemple, elle n'est pas simple car elle a une auto-intersection.

```
SELECT ST_IsSimple(ST_GeomFromText('LINESTRING(0 0,10 0,5 5,5 -5)'));
```

```
 st_issimple
-----------
 false
```

ST_ IsValid

ST_ IsValid renvoie true si la projection 2D de la géométrie d'entrée est valide. Pour plus d'informations sur la définition d'une géométrie valide, consultez Validité géométrique.

Syntaxe

```
ST_IsValid(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

BOOLEAN

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant vérifie si le polygone spécifié est valide. Dans cet exemple, le polygone n'est pas valide car l'intérieur du polygone n'est pas simplement connecté.

```
SELECT ST_IsValid(ST_GeomFromText('POLYGON((0 0,10 0,10 10,0 10,0 0),(5 0,10 5,5 10,0
 5,5 0))'));
```

```
 st_isvalid
------------
 false
```

ST_Length

Pour une géométrie linéaire, ST_Length renvoie la longueur cartésienne d'une projection 2D. Les unités de longueur sont les mêmes que les unités dans lesquelles les coordonnées de la géométrie en entrée sont exprimées. La fonction renvoie zéro (0) pour les points, les multipoints et les géométries surfaciques. Lorsque l'entrée est une collection de géométries, la fonction renvoie la somme des longueurs des géométries de la collection.

Pour une géographie, ST_Length renvoie la longueur géodésique de la projection 2D d'une géographie linéaire d'entrée calculée sur le sphéroïde déterminé par le SRID. L'unité de longueur est exprimée en mètres. La fonction renvoie zéro (0) pour les points, les multipoints et les géographies surfaciques. Lorsque l'entrée est une collection de géométries, la fonction renvoie la somme des longueurs des géographies de la collection.

Syntaxe

```
ST_Length(geo)
```

Arguments

geo

Valeur de type de données GEOMETRY ou GEOGRAPHY ou expression qui est évaluée sur un type GEOMETRY ou GEOGRAPHY.

Type de retour

DOUBLE PRECISION

Si geo est null, null est renvoyé.

Si la valeur SRID est introuvable, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie la longueur cartésienne d'une multilinestring.

```
SELECT ST_Length(ST_GeomFromText('MULTILINESTRING((0 0,10 0,0 10),(10 0,20 0,20
 10))'));
```

```
st_length
--------------------------------
  44.142135623731
```

Le code SQL suivant renvoie la longueur d'une linestring dans une géographie.

```
SELECT ST_Length(ST_GeogFromText('SRID=4326;LINESTRING(5 0,6 0,4 0)'));
```

```
 st_length
------------------
 333958.472379804
```

Le code SQL suivant renvoie la longueur d'un point dans une géographie.

```
SELECT ST_Length(ST_GeogFromText('SRID=4326;POINT(4 5)'));
```

```
 st_length
-----------
 0
```

ST_ LengthSphere

ST_ LengthSphere renvoie la longueur d'une géométrie linéaire en mètres. Pour les géométries ponctuelles, multipoints et surfaciques, LengthSphere ST_ renvoie 0. Pour les collections de géométries, ST_ LengthSphere renvoie la longueur totale des géométries linéaires de la collection en mètres.

ST_ LengthSphere interprète les coordonnées de chaque point de la géométrie en entrée sous forme de longitude et de latitude en degrés. Pour les géométries 3DZ, 3DM ou 4D, seules les deux premières coordonnées sont utilisées.

Syntaxe

```
ST_LengthSphere(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

Longueur `DOUBLE PRECISION` en mètres. Le calcul de la longueur est basé sur le modèle sphérique de la Terre dont le rayon est le rayon moyen de la Terre du modèle ellipsoïdal World Geodetic System (WGS) 84 de la Terre.

Si geom est null, null est renvoyé.

Exemples

L'exemple de code SQL suivant calcule la longueur d'une linestring en mètres.

```
SELECT ST_LengthSphere(ST_GeomFromText('LINESTRING(10 10,45 45)'));
```

```
 st_lengthsphere
------------------
 5127736.08292556
```

ST_Length2D

ST_Length2D est un alias pour ST_Length. Pour de plus amples informations, veuillez consulter ST_Length.

ST_ LineFromMultiPoint

ST_ LineFromMultiPoint renvoie une chaîne linéaire à partir d'une géométrie multipoint en entrée. L'ordre des points est préservé. La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID des géométries d'entrée. La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Syntaxe

```
ST_LineFromMultiPoint(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Le sous-type doit être MULTIPOINT.

Type de retour

GEOMETRY

Si geom est null, null est renvoyé.

Si geom est vide, un LINESTRING vide est renvoyé.

Si geom contient des points vides, ceux-ci sont ignorés.

Si geom n'est pas un MULTIPOINT, une erreur est renvoyée.

Exemples

Le code SQL suivant crée une linestring à partir d'un multipoint.

```
SELECT ST_AsEWKT(ST_LineFromMultiPoint(ST_GeomFromText('MULTIPOINT(0 0,10 0,10 10,5 5,0
 5)',4326)));
```

```
 st_asewkt
-------------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 5,0 5)
```

ST_ LineInterpolatePoint

ST_ LineInterpolatePoint renvoie un point le long d'une ligne à une distance fractionnaire du début de la ligne.

Pour déterminer l'égalité des points, ST_ LineInterpolatePoint agit sur la projection 2D de la géométrie d'entrée. Si la géométrie en entrée est vide, une copie de celle-ci est renvoyée dans la même dimension que l'entrée. Pour les géométries 3DZ, 3DM et 4D, les coordonnées `z` ou `m` sont la moyenne des coordonnées `z` ou `m` du segment où se trouve le point.

Syntaxe

```
ST_LineInterpolatePoint(geom, fraction)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type est `LINESTRING`.

fraction

Valeur du type de données `DOUBLE PRECISION` qui représente la position d'un point le long de la linestring pour la ligne. La valeur est une fraction comprise entre 0 et 1, inclus.

Type de retour

`GEOMETRY` du sous-type `POINT`.

Si geom ou fraction est null, null est renvoyé.

Si geom est vide, le point vide est renvoyé.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID de la géométrie d'entrée.

Si fraction est hors limites, une erreur est renvoyée.

Si geom n'est pas une linestring, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie un point à mi-chemin d'une linestring.

```
SELECT ST_AsEWKT(ST_LineInterpolatePoint(ST_GeomFromText('LINESTRING(0 0, 5 5, 7 7, 10
 10)'), 0.50));
```

```
st_asewkt
-----------
 POINT(5 5)
```

Le code SQL suivant renvoie un point à 90 % du chemin le long d'une linestring.

```
SELECT ST_AsEWKT(ST_LineInterpolatePoint(ST_GeomFromText('LINESTRING(0 0, 5 5, 7 7, 10
 10)'), 0.90));
```

```
st_asewkt
-----------
 POINT(9 9)
```

ST_M

ST_M renvoie la coordonnée `m` d'un point d'entrée.

Syntaxe

```
ST_M(point)
```

Arguments

point

Valeur `POINT` du type de données `GEOMETRY`.

Type de retour

Valeur DOUBLE PRECISION de la coordonnée m.

If point est null, null est renvoyé.

Si point est un point 2D ou 3DZ, null est renvoyé.

Si point est le point vide, null est renvoyé.

Si point n'est pas un POINT, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie la coordonnée m d'un point dans une géométrie 3DM.

```
SELECT ST_M(ST_GeomFromEWKT('POINT M (1 2 3)'));
```

```
st_m
-----------
 3
```

Le SQL suivant renvoie la coordonnée m d'un point dans une géométrie 4D.

```
SELECT ST_M(ST_GeomFromEWKT('POINT ZM (1 2 3 4)'));
```

```
st_m
-----------
 4
```

ST_ MakeEnvelope

ST_ MakeEnvelope renvoie une géométrie comme suit :

- Si les coordonnées en entrée spécifient un point, la géométrie renvoyée est un point.
- Si les coordonnées en entrée spécifient une ligne, la géométrie renvoyée est une linestring.
- Sinon, la géométrie renvoyée est un polygone, où les coordonnées en entrée spécifient les coins inférieur gauche et supérieur droit d'un cadre.

Si elle est fournie, la valeur de l'identificateur de système de référence spatiale (SRID) de la géométrie renvoyée est définie sur la valeur SRID d'entrée.

Syntaxe

```
ST_MakeEnvelope(xmin, ymin, xmax, ymax)
```

```
ST_MakeEnvelope(xmin, ymin, xmax, ymax, srid)
```

Arguments

xmin

Valeur du type de données `DOUBLE PRECISION`. Cette valeur est la première coordonnée du coin inférieur gauche d'un cadre.

ymin

Valeur du type de données `DOUBLE PRECISION`. Cette valeur est la deuxième coordonnée du coin inférieur gauche d'un cadre.

xmax

Valeur du type de données `DOUBLE PRECISION`. Cette valeur est la première coordonnée du coin supérieur droit d'un cadre.

ymax

Valeur du type de données `DOUBLE PRECISION`. Cette valeur est la deuxième coordonnée du coin supérieur droit d'un cadre.

srid

Valeur de type de données `INTEGER` qui représente un identificateur de système de référence spatiale (SRID). Si la valeur SRID n'est pas fournie, elle est définie sur zéro.

Type de retour

`GEOMETRY` de sous-type `POINT`, `LINESTRING` ou `POLYGON`.

Le SRID de la géométrie renvoyée est défini sur `srid` ou zéro si `srid` n'est pas défini.

Si xmin, ymin, xmax, ymax ou srid est null, alors null est renvoyé.

Si srid est négatif, une erreur est renvoyée.

Exemples

Le SQL suivant renvoie un polygone représentant une enveloppe définie par les quatre valeurs de coordonnées en entrée.

```
SELECT ST_AsEWKT(ST_MakeEnvelope(2,4,5,7));
```

```
 st_astext
---------------
 POLYGON((2 4,2 7,5 7,5 4,2 4))
```

Le code SQL suivant renvoie un polygone représentant une enveloppe définie par les quatre valeurs de coordonnées en entrée et une valeur SRID.

```
SELECT ST_AsEWKT(ST_MakeEnvelope(2,4,5,7,4326));
```

```
 st_astext
---------------------------------
 SRID=4326;POLYGON((2 4,2 7,5 7,5 4,2 4))
```

ST_ MakeLine

ST_ MakeLine crée une chaîne de lignes à partir des géométries d'entrée.

La dimension de la géométrie renvoyée est identique à celle des géométries en entrée. Les deux géométries en entrée doivent avoir la même dimension.

Syntaxe

```
ST_MakeLine(geom1, geom2)
```

Arguments

geom1

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type doit être `POINT`, `LINESTRING` ou `MULTIPOINT`.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Le sous-type doit être POINT, LINESTRING ou MULTIPOINT.

Type de retour

GEOMETRY du sous-type LINESTRING.

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 et geom2 sont des points vides ou contiennent des points vides, ces points vides sont ignorés.

Si geom1 et geom2 sont vides, la LINESTRING vide est renvoyée.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID des géométries d'entrée.

Si geom1 et geom2 ont des valeurs SRID différentes, une erreur est renvoyée.

Si geom1 ou geom2 n'est pas un POINT, LINESTRING, ou MULTIPOINT, alors une erreur est renvoyée.

Si geom1 et geom2 ont des dimensions différentes, une erreur est renvoyée.

Exemples

Le SQL suivant construit une linestring à partir de deux linestrings d'entrée.

```
SELECT ST_MakeLine(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'), ST_GeomFromText('LINESTRING(88.29 39.07,88.42 39.26,88.27
 39.31,88.29 39.07)'));
```

```
st_makeline
-----------
 0102000000080000000C3F5285C8F52534052B81E85EB113D407B14AE47E15A5340C3F5285C8F423D40E17A14AE4751
```

ST_ MakePoint

ST_ MakePoint renvoie une géométrie ponctuelle dont les valeurs de coordonnées sont les valeurs d'entrée.

Syntaxe

```
ST_MakePoint(x, y)
```

```
ST_MakePoint(x, y, z)
```

```
ST_MakePoint(x, y, z, m)
```

Arguments

h/24, j/7

Valeur du type de données `DOUBLE PRECISION` représentant la première coordonnée.

y

Valeur du type de données `DOUBLE PRECISION` représentant la deuxième coordonnée.

z

Valeur du type de données `DOUBLE PRECISION` représentant la troisième coordonnée.

m

Valeur du type de données `DOUBLE PRECISION` représentant la quatrième coordonnée.

Type de retour

`GEOMETRY` du sous-type `POINT`.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est définie sur 0.

Si x, y, z ou m est null, alors null est renvoyé.

Exemples

Le SQL suivant renvoie un type `GEOMETRY` de sous-type `POINT` avec les coordonnées fournies.

```
SELECT ST_AsText(ST_MakePoint(1,3));
```

```
st_astext
-----------
 POINT(1 3)
```

Le SQL suivant renvoie un type GEOMETRY de sous-type POINT avec les coordonnées fournies.

```
SELECT ST_AsEWKT(ST_MakePoint(1, 2, 3));
```

```
st_asewkt
----------------
 POINT Z (1 2 3)
```

Le SQL suivant renvoie un type GEOMETRY de sous-type POINT avec les coordonnées fournies.

```
SELECT ST_AsEWKT(ST_MakePoint(1, 2, 3, 4));
```

```
st_asewkt
-------------------
 POINT ZM (1 2 3 4)
```

ST_ MakePolygon

ST_ MakePolygon possède deux variantes qui renvoient un polygone. L'une prend une géométrie unique et l'autre prend deux géométries.

- L'entrée de la première variante est une linestring qui définit l'anneau externe du polygone en sortie.
- L'entrée de la deuxième variante est une linestring et une multilinestring. Les deux sont vides ou fermées.

  La limite de l'anneau extérieur du polygone en sortie est la linestring en entrée, et les limites des anneaux intérieurs du polygone sont les linestrings dans la multilinestring d'entrée. Si la linestring

d'entrée est vide, un polygone vide est renvoyé. Les linestrings vides dans la multilinestring sont ignorées. L'identifiant du système de référence spatiale (SRID) de la géométrie résultante est le SRID commun des deux géométries d'entrée.

La dimension de la géométrie renvoyée est identique à celle des géométries en entrée. L'anneau extérieur et les anneaux intérieurs doivent avoir la même dimension.

Syntaxe

```
ST_MakePolygon(geom1)
```

```
ST_MakePolygon(geom1, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Le sous-type doit être LINESTRING. La valeur linestring doit être fermée ou vide.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Le sous-type doit être MULTILINESTRING.

Type de retour

GEOMETRY du sous-type POLYGON.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est égale au SRID des entrées.

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 n'est pas une linestring, une erreur est renvoyée.

Si geom2 n'est pas une multilinestring, une erreur est renvoyée.

Si geom1 n'est pas fermé, une erreur est renvoyée.

Si geom1 n’est pas un point unique ou n’est pas fermé, une erreur est renvoyée.

Si geom2 contient au moins une linestring qui a un point unique ou n’est pas fermée, une erreur est renvoyée.

Si geom1 et geom2 ont des valeurs SRID différentes, une erreur est renvoyée.

Si geom1 et geom2 ont des dimensions différentes, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie un polygone depuis une linestring d’entrée.

```
SELECT ST_AsText(ST_MakePolygon(ST_GeomFromText('LINESTRING(77.29 29.07,77.42
 29.26,77.27 29.31,77.29 29.07)')));
```

```
 st_astext
---------------
POLYGON((77.29 29.07,77.42 29.26,77.27 29.31,77.29 29.07))
```

Le code SQL suivant crée un polygone à partir d’une linestring fermée et d’un multilinestring fermée. La linestring est utilisée pour l’anneau extérieur du polygone. Les linestrings dans les multilinestrings sont utilisées pour les anneaux intérieurs du polygone.

```
SELECT ST_AsEWKT(ST_MakePolygon(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,0 10,0 0)'),
 ST_GeomFromText('MULTILINESTRING((1 1,1 2,2 1,1 1),(3 3,3 4,4 3,3 3))')));
```

```
 st_astext
---------------------------------
POLYGON((0 0,10 0,10 10,0 10,0 0),(1 1,1 2,2 1,1 1),(3 3,3 4,4 3,3 3))
```

ST_ MemSize

ST_ MemSize renvoie la quantité d'espace mémoire (en octets) utilisée par la géométrie d'entrée. Cette taille dépend de la représentation Amazon Redshift interne de la géométrie et peut donc changer si la représentation interne change. Vous pouvez utiliser cette taille comme indication de la taille relative des objets géométriques dans Amazon Redshift.

Syntaxe

```
ST_MemSize(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

INTEGER représentant la dimension inhérente de geom.

Si geom est null, null est renvoyé.

Exemples

Le SQL suivant renvoie la taille de mémoire d'une collection de géométries.

```
SELECT ST_MemSize(ST_GeomFromText('GEOMETRYCOLLECTION(POLYGON((0 0,10 0,0 10,0
 0)),LINESTRING(20 10,20 0,10 0))')::varchar + ' bytes';
```

```
 ?column?
-----------
 172 bytes
```

ST_ MMax

ST_ MMax renvoie la m coordonnée maximale d'une géométrie en entrée.

Syntaxe

```
ST_MMax(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

Valeur `DOUBLE PRECISION` de la coordonnée `m` maximum.

Si geom est vide, null est renvoyé.

Si geom est null, null est renvoyé.

Si geom est une géométrie 2D ou 3DZ, null est renvoyé.

Exemples

Le code SQL suivant renvoie la coordonnée `m` la plus importante d'une linestring dans une géométrie 3DM.

```
SELECT ST_MMax(ST_GeomFromEWKT('LINESTRING M (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_mmax
-----------
  8
```

Le code SQL suivant renvoie la coordonnée `m` la plus importante d'une linestring dans une géométrie 4D.

```
SELECT ST_MMax(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_mmax
-----------
  11
```

ST_ MMin

ST_ MMin renvoie la `m` coordonnée minimale d'une géométrie en entrée.

Syntaxe

```
ST_MMin(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

Valeur `DOUBLE PRECISION` de la coordonnée `m` minimum.

Si geom est vide, null est renvoyé.

Si geom est null, null est renvoyé.

Si geom est une géométrie 2D ou 3DZ, null est renvoyé.

Exemples

Le code SQL suivant renvoie la coordonnée `m` la moins importante d'une linestring dans une géométrie 3DM.

```
SELECT ST_MMin(ST_GeomFromEWKT('LINESTRING M (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_mmin
-----------
  2
```

Le code SQL suivant renvoie la coordonnée `m` la moins importante d'une linestring dans une géométrie 4D.

```
SELECT ST_MMin(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_mmin
-----------
  3
```

ST_Multi

ST_multi convertit une géométrie en multitype correspondant. Si la géométrie en entrée est déjà un multitype ou une collection de géométries, une copie de celle-ci est renvoyée. Si la géométrie en entrée est un point, une linestring ou un polygone, alors un multipoint, une multilinestring ou un multipolygone (respectivement) contenant la géométrie en entrée est renvoyé.

Syntaxe

```
ST_Multi(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`GEOMETRY` avec sous-type `MULTIPOINT`, `MULTILINESTRING`, `MULTIPOLYGON` ou `GEOMETRYCOLLECTION`.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID des géométries d'entrée.

Si geom est null, null est renvoyé.

Exemples

Le SQL suivant renvoie un multipoint depuis un multipoint en entrée.

```
SELECT ST_AsEWKT(ST_Multi(ST_GeomFromText('MULTIPOINT((1 2),(3 4))', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;MULTIPOINT((1 2),(3 4))
```

Le code SQL suivant renvoie un multipoint depuis un point d'entrée.

```
SELECT ST_AsEWKT(ST_Multi(ST_GeomFromText('POINT(1 2)', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;MULTIPOINT((1 2))
```

Le code SQL suivant renvoie une collection de géométries depuis une collection d'entrées.

```
SELECT ST_AsEWKT(ST_Multi(ST_GeomFromText('GEOMETRYCOLLECTION(POINT(1 2),MULTIPOINT((1
 2),(3 4)))', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;GEOMETRYCOLLECTION(POINT(1 2),MULTIPOINT((1 2),(3 4)))
```

ST_ NDims

ST_ NDims renvoie la dimension des coordonnées d'une géométrie. ST_ NDims ne prend pas en compte la dimension topologique d'une géométrie. Au lieu de cela, il renvoie une valeur constante en fonction de la dimension de la géométrie.

Syntaxe

```
ST_NDims(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`INTEGER` représentant la dimension inhérente de geom.

Si geom est null, null est renvoyé.

Les valeurs renvoyées sont comme suit :

Valeur renvoyée	Dimension de la géométrie en entrée
2	2D
3	3DZ ou 3DM
4	4D

Exemples

Le code SQL suivant renvoie le nombre de dimensions d'une linestring 2D.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING(0 0,1 1,2 2,0 0)'));
```

```
st_ndims
-------------
 2
```

Le code SQL suivant renvoie le nombre de dimensions d'une linestring 3DZ.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING Z(0 0 3,1 1 3,2 2 3,0 0 3)'));
```

```
st_ndims
-------------
 3
```

Le code SQL suivant renvoie le nombre de dimensions d'une linestring 3DM.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING M(0 0 4,1 1 4,2 2 4,0 0 4)'));
```

```
st_ndims
-------------
 3
```

Le code SQL suivant renvoie le nombre de dimensions d'une linestring 4D.

```
SELECT ST_NDims(ST_GeomFromText('LINESTRING ZM(0 0 3 4,1 1 3 4,2 2 3 4,0 0 3 4)'));
```

```
st_ndims
-------------
 4
```

ST_ NPoints

ST_ NPoints renvoie le nombre de points non vides dans une géométrie ou une géographie en entrée.

Syntaxe

```
ST_NPoints(geo)
```

Arguments

geo

Valeur de type de données GEOMETRY ou GEOGRAPHY ou expression qui est évaluée sur un type GEOMETRY ou GEOGRAPHY.

Type de retour

INTEGER

Si geo est un point vide, 0 est renvoyé.

Si geo est null, null est renvoyé.

Exemples

Le code SQL suivant renvoie le nombre de points d'une linestring.

```
SELECT ST_NPoints(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_npoints
```

```
-------------
 4
```

Le code SQL suivant renvoie le nombre de points d'une linestring dans une géographie.

```
SELECT ST_NPoints(ST_GeogFromText('LINESTRING(110 40, 2 3, -10 80, -7 9)'));
```

```
st_npoints
-------------
 4
```

ST_ NRings

ST_ NRings renvoie le nombre d'anneaux dans une géométrie d'entrée.

Syntaxe

```
ST_NRings(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`INTEGER`

Si geom est null, null est renvoyé.

Les valeurs renvoyées sont comme suit :

Valeur renvoyée	Sous-type de géométrie
0	Renvoyé si geom est un sous-type `POINT`, `LINESTRING` , `MULTIPOINT` ou `MULTILINESTRING` .

Valeur renvoyée	Sous-type de géométrie
Le nombre d'anneaux.	Renvoyé si geom est un sous-type POLYGON ou MULTIPOLYGON .
Le nombre d'anneaux dans tous les composants.	Renvoyé si geom est un sous-type GEOMETRYCOLLECTION

Exemples

Le SQL suivant renvoie le nombre d'anneaux dans un multipolygone.

```
SELECT ST_NRings(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((0 0,-10 0,0
 -10,0 0)))'));
```

```
 st_nrings
-------------
 2
```

ST_ NumGeometries

ST_ NumGeometries renvoie le nombre de géométries d'une géométrie d'entrée.

Syntaxe

```
ST_NumGeometries(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

INTEGER représentant le nombre de géométries dans geom.

Si geom est null, null est renvoyé.

Si geom est une géométrie vide unique, `0` est renvoyé.

Si geom est une géométrie non vide unique, 1 est renvoyé.

Si geom est un sous-type `GEOMETRYCOLLECTION` ou `MULTI`, le nombre de géométries est renvoyé.

Exemples

Le code SQL suivant renvoie le nombre de géométries dans la multilinestring d'entrée.

```
SELECT ST_NumGeometries(ST_GeomFromText('MULTILINESTRING((0 0,1 0,0 5),(3 4,13 26))'));
```

```
st_numgeometries
-------------
 2
```

ST_ NumInteriorRings

ST_ NumInteriorRings renvoie le nombre d'anneaux dans une géométrie polygonale en entrée.

Syntaxe

```
ST_NumInteriorRings(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`INTEGER`

Si geom est null, null est renvoyé.

Si geom n'est pas un polygone, null est renvoyé.

Exemples

Le SQL suivant renvoie le nombre de géométries d'anneaux à l'intérieur du polygone d'entrée.

```
SELECT ST_NumInteriorRings(ST_GeomFromText('POLYGON((0 0,100 0,100 100,0 100,0 0),(1
 1,1 5,5 1,1 1),(7 7,7 8,8 7,7 7))'));
```

```
 st_numinteriorrings
-------------
 2
```

ST_ NumPoints

ST_ NumPoints renvoie le nombre de points dans une géométrie d'entrée.

Syntaxe

```
ST_NumPoints(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`INTEGER`

Si geom est null, null est renvoyé.

Si geom n'est pas de sous-type `LINESTRING`, la valeur null est renvoyée.

Exemples

Le code SQL suivant renvoie le nombre de points d'une linestring d'entrée.

```
SELECT ST_NumPoints(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'));
```

```
st_numpoints
-------------
```

```
4
```

Le SQL suivant renvoie la valeur null car geom en entrée n'est pas de sous-type `LINESTRING`.

```
SELECT ST_NumPoints(ST_GeomFromText('MULTIPOINT(1 2,3 4)'));
```

```
st_numpoints
-------------
```

ST_Perimeter

Pour une géométrie surfacique, ST_Perimeter renvoie le périmètre cartésien (longueur de la frontière) de la projection 2D. Les unités de périmètre sont les mêmes que les unités dans lesquelles les coordonnées de la géométrie en entrée sont exprimées. La fonction renvoie zéro (0) pour les points, les multipoints et les géométries linéaires. Lorsque l'entrée est une collection de géométries, la fonction renvoie la somme des périmètres des géométries de la collection.

Pour une géographie d'entrée, ST_Perimeter renvoie le périmètre géodésique (longueur de la limite) de la projection 2D d'une géographie surfacique d'entrée calculée sur le sphéroïde déterminé par le SRID. L'unité de périmètre est exprimée en mètres. La fonction renvoie zéro (0) pour les points, les multipoints et les géomgraphies linéaires. Lorsque l'entrée est une collection de géométries, la fonction renvoie la somme des périmètres des géographies de la collection.

Syntaxe

```
ST_Perimeter(geo)
```

Arguments

geo

Valeur de type de données `GEOMETRY` ou `GEOGRAPHY` ou expression qui est évaluée sur un type `GEOMETRY` ou `GEOGRAPHY`.

Type de retour

`DOUBLE PRECISION`

Si geo est null, null est renvoyé.

Si la valeur SRID est introuvable, une erreur est renvoyée.

Exemples

Le SQL suivant renvoie le périmètre cartésien d'un multipolygone.

```
SELECT ST_Perimeter(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((10 0,20 0,20
 10,10 0)))'));
```

```
 st_perimeter
------------------------------
    68.2842712474619
```

Le SQL suivant renvoie le périmètre cartésien d'un multipolygone.

```
SELECT ST_Perimeter(ST_GeomFromText('MULTIPOLYGON(((0 0,10 0,0 10,0 0)),((10 0,20 0,20
 10,10 0)))'));
```

```
 st_perimeter
------------------------------
    68.2842712474619
```

Le code SQL suivant renvoie le périmètre d'un polygone dans une géographie.

```
SELECT ST_Perimeter(ST_GeogFromText('SRID=4326;POLYGON((0 0,1 0,0 1,0 0))'));
```

```
 st_perimeter
------------------
 378790.428393693
```

Le code SQL suivant renvoie le périmètre d'une linestring dans une géographie.

```
SELECT ST_Perimeter(ST_GeogFromText('SRID=4326;LINESTRING(5 0,10 0)'));
```

```
 st_perimeter
--------------
 0
```

ST_Perimeter2D

ST_Perimeter2D est un alias pour ST_Perimeter. Pour plus d'informations, consultez ST_Perimeter.

ST_Point

ST_Point renvoie une géométrie de point à partir des valeurs des coordonnées d’entrée.

Syntaxe

```
ST_Point(x, y)
```

Arguments

h/24, j/7

Valeur du type de données `DOUBLE PRECISION` représentant la première coordonnée.

y

Valeur du type de données `DOUBLE PRECISION` représentant la deuxième coordonnée.

Type de retour

`GEOMETRY` du sous-type `POINT`.

La valeur de l’identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est définie sur 0.

Si x ou y est null, null est renvoyé.

Exemples

Le SQL suivant construit une géométrie de point à partir des coordonnées d’entrée.

```
SELECT ST_AsText(ST_Point(5.0, 7.0));
```

```
st_astext
-------------
POINT(5 7)
```

ST_PointN

ST_PointN renvoie un point dans une linestring comme spécifié par une valeur d'index. Les valeurs d'index négatives sont comptées à l'envers à partir de la fin de la linestring, de sorte que -1 est le dernier point.

La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Syntaxe

```
ST_PointN(geom, index)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type doit être `LINESTRING`.

index

Valeur du type de données `INTEGER` qui représente l'index d'un point dans une linestring.

Type de retour

`GEOMETRY` du sous-type `POINT`.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est définie sur 0.

Si geom ou index est null, null est renvoyé.

Si l'index est hors de portée, null est renvoyé.

Si geom est vide, null est renvoyé.

Si geom n'est pas un `LINESTRING`, null est renvoyé.

Exemples

Le SQL suivant renvoie une représentation de texte connu étendu (EWKT) d'un `LINESTRING` à six points à un objet `GEOMETRY`, et renvoie le point à l'index 5 de la linestring.

```
SELECT ST_AsEWKT(ST_PointN(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5,0
 0)',4326), 5));
```

```
st_asewkt
-------------
 SRID=4326;POINT(0 5)
```

ST_Points

ST_Points renvoie une géométrie multipoint contenant tous les points non vides de la géométrie en entrée. ST_Points ne supprime pas les points dupliqués dans l'entrée, y compris les points de début et de fin des géométries en anneau.

Syntaxe

```
ST_Points(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`GEOMETRY` du sous-type `MULTIPOINT`.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est identique à celle de geom.

Si geom est null, null est renvoyé.

Si geom est vide, le multipoint vide est renvoyé.

Exemples

Les exemples SQL suivants construisent une géométrie multipoint à partir de la géométrie en entrée. Le résultat est une géométrie multipoint contenant les points non vides de la géométrie en entrée.

```
SELECT ST_AsEWKT(ST_Points(ST_SetSRID(ST_GeomFromText('LINESTRING(1 0,2 0,3 0)'),
 4326)));
```

```
st_asewkt
-------------
SRID=4326;MULTIPOINT((1 0),(2 0),(3 0))
```

```
SELECT ST_AsEWKT(ST_Points(ST_SetSRID(ST_GeomFromText('MULTIPOLYGON(((0 0,1 0,0 1,0
 0)))'), 4326)));
```

```
st_asewkt
-------------
SRID=4326;MULTIPOINT((0 0),(1 0),(0 1),(0 0))
```

ST_Polygon

ST_Polygon renvoie une géométrie de polygone dont l'anneau extérieur est la linestring d'entrée avec la valeur qui a été entrée pour l'identifiant de système de référence spatiale (SRID).

La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Syntaxe

```
ST_Polygon(linestring, srid)
```

Arguments

linestring

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`. Le sous-type doit être `LINESTRING` qui représente une linestring. La valeur linestring doit être fermée.

srid

Valeur du type de données `INTEGER` représentant un SRID.

Type de retour

`GEOMETRY` du sous-type `POLYGON`.

La valeur de SRID de la géométrie renvoyée est définie sur srid.

Si linestring ou srid est null, null est renvoyé.

Si linestring n'est pas une linestring, une erreur est renvoyée.

Si linestring n'est pas fermé, une erreur est renvoyée.

Si srid est négatif, une erreur est renvoyée.

Exemples

Le SQL suivant construit un polygone avec une valeur SRID.

```
SELECT ST_AsEWKT(ST_Polygon(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'),4356));
```

```
st_asewkt
-------------
 SRID=4356;POLYGON((77.29 29.07,77.42 29.26,77.27 29.31,77.29 29.07))
```

ST_ RemovePoint

ST_ RemovePoint renvoie une géométrie de chaîne de lignes dont le point de la géométrie d'entrée situé à une position d'index a été supprimé.

L'index est basé sur zéro. L'identificateur du système de référence spatiale (SRID) du résultat est identique à la géométrie en entrée. La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Syntaxe

```
ST_RemovePoint(geom, index)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Le sous-type doit être LINESTRING.

index

Valeur de type de données INTEGER qui représente la position d'un index de base zéro.

Type de retour

GEOMETRY

Si geom ou index est null, null est renvoyé.

Si geom n'est pas un sous-type LINESTRING, une erreur est renvoyée.

Si l'index est hors de portée, une erreur est renvoyée. Les valeurs valides pour la position d'index sont comprises entre 0 et ST_NumPoints(geom) moins 1.

Exemples

Le code SQL suivant supprime le dernier point d'une linestring.

```
WITH tmp(g) AS (SELECT ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0 5)',4326))
SELECT ST_AsEWKT(ST_RemovePoint(g, ST_NumPoints(g) - 1)) FROM tmp;
```

```
   st_asewkt
----------------------------------------
 SRID=4326;LINESTRING(0 0,10 0,10 10,5 5)
```

ST_Reverse

ST_reverse inverse l'ordre des sommets pour les géométries linéaires et surfaciques. Pour les géométries de point ou multipoint, une copie de la géométrie d'origine est renvoyée. Pour les collections de géométrie, ST_Reverse inverse l'ordre des sommets de chacune des géométries de la collection.

La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Syntaxe

```
ST_Reverse(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

GEOMETRY

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID des géométries d'entrée.

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant inverse l'ordre des points dans une linestring.

```
SELECT ST_AsEWKT(ST_Reverse(ST_GeomFromText('LINESTRING(1 0,2 0,3 0,4 0)', 4326)));
```

```
    st_asewkt
-----------------------------------
  SRID=4326;LINESTRING(4 0,3 0,2 0,1 0)
```

ST_ SetPoint

ST_ SetPoint renvoie une chaîne de lignes avec des coordonnées mises à jour par rapport à la position de la chaîne de ligne d'entrée telle que spécifiée par l'index. Les nouvelles coordonnées sont les coordonnées du point d'entrée.

La dimension de la géométrie renvoyée est identique à celle de la valeur geom1. Si geom1 et geom2 ont des dimensions différentes, geom2 est projeté à la dimension de geom1.

Syntaxe

```
ST_SetPoint(geom1, index, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Le sous-type doit être LINESTRING.

index

Valeur de type de données INTEGER qui représente la position d'un index. Un 0 fait référence au premier point de la linestring à partir de la gauche, un 1 fait référence au deuxième point, et ainsi de suite. L'index peut être une valeur négative. Un -1 fait référence au premier point de la linestring à partir de la droite, un -2 fait référence au deuxième point, et ainsi de suite.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Le sous-type doit être POINT.

Type de retour

GEOMETRY

Si geom2 est le point vide, geom1 est renvoyé.

Si geom1, geom2, ou index est null, alors null est renvoyé.

Si geom1 n'est pas une linestring, une erreur est renvoyée.

Si index n'est pas dans une plage d'index valide, une erreur est renvoyée.

Si geom2 n'est pas un point, une erreur est renvoyée.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie une nouvelle linestring où nous définissons le deuxième point de la linestring d'entrée avec le point spécifié.

```
SELECT ST_AsText(ST_SetPoint(ST_GeomFromText('LINESTRING(1 2, 3 2, 5 2, 1 2)'), 2,
 ST_GeomFromText('POINT(7 9)')));
```

```
st_astext
-------------
 LINESTRING(1 2,3 2,7 9,1 2)
```

L'exemple de code SQL suivant renvoie une nouvelle linestring où nous définissons le troisième point à partir de la droite (l'index est négatif) de la linestring avec le point spécifié.

```
SELECT ST_AsText(ST_SetPoint(ST_GeomFromText('LINESTRING(1 2, 3 2, 5 2, 1 2)'), -3,
 ST_GeomFromText('POINT(7 9)')));
```

```
st_astext
-------------
 LINESTRING(1 2,7 9,5 2,1 2)
```

ST_SetSRID

ST_SetSRID renvoie une géométrie identique à la géométrie d'entrée, mais mise à jour avec la valeur entrée pour l'identifiant de système de référence spatiale (SRID).

Syntaxe

```
ST_SetSRID(geom, srid)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

srid

Valeur du type de données `INTEGER` représentant un SRID.

Type de retour

`GEOMETRY`

La valeur de SRID de la géométrie renvoyée est définie sur srid.

Si geom ou srid est null, null est renvoyé.

Si srid est négatif, une erreur est renvoyée.

Exemples

Le code SQL suivant définit la valeur SRID d'une linestring.

```
SELECT ST_AsEWKT(ST_SetSRID(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27
 29.31,77.29 29.07)'),50));
```

```
st_asewkt
-------------
 SRID=50;LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29 29.07)
```

ST_Simplify

ST_Simplify renvoie une copie simplifiée de la géométrie d'entrée à l'aide de l' Ramer-Douglas-Peuckeralgorithme avec la tolérance donnée. Il est possible que la topologie de la géométrie en entrée ne soit pas conservée. Pour plus d'informations sur cet algorithme, consultez Algorithme de Douglas-Peucker sur Wikipédia.

Lorsque ST_Simplify calcule les distances pour simplifier une géométrie, ST_Simplify fonctionne sur la projection 2D de la géométrie en entrée.

Syntaxe

```
ST_Simplify(geom, tolerance)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

tolerance

Valeur de type de données `DOUBLE PRECISION` qui représente le niveau de tolérance de l' Ramer-Douglas-Peuckeralgorithme. Si tolerance est un nombre négatif, zéro est utilisé.

Type de retour

`GEOMETRY`.

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID de la géométrie d'entrée.

La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant simplifie la chaîne de ligne d'entrée en utilisant une tolérance de distance euclidienne de 1 avec l'algorithme. Ramer-Douglas-Peucker Les unités de distance sont identiques à celles indiquées dans les coordonnées de la géométrie.

```
SELECT ST_AsEWKT(ST_Simplify(ST_GeomFromText('LINESTRING(0 0,1 2,1 1,2 2,2 1)'), 1));
```

```
 st_asewkt
-----------
LINESTRING(0 0,1 2,2 1)
```

ST_SRID

ST_SRID renvoie l'identifiant de système de référence spatiale (SRID) d'une géométrie d'entrée. Pour plus d'informations sur un SRID, consultez Interrogation des données spatiales dans Amazon Redshift.

Syntaxe

```
ST_SRID(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

`INTEGER` représentant la valeur SRID de geom.

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant renvoie une valeur SRID d'une linestring définie sur un SRID de `4326`.

```
SELECT ST_SRID(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)',4326));
```

```
st_srid
-------------
 4326
```

Le code SQL suivant renvoie une valeur SRID d'une linestring qui n'est pas définie à sa construction. Cela se traduit par `0` pour la valeur SRID.

```
SELECT ST_SRID(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_srid
-------------
 0
```

ST_ StartPoint

ST_ StartPoint renvoie le premier point d'une chaîne de ligne d'entrée. La valeur SRID (Spatial Reference System Identifier) du résultat est identique à celle de la géométrie en entrée. La dimension de la géométrie renvoyée est identique à celle de la géométrie en entrée.

Syntaxe

```
ST_StartPoint(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Le sous-type doit être LINESTRING.

Type de retour

GEOMETRY

Si geom est null, null est renvoyé.

Si geom est vide, null est renvoyé.

Si geom n'est pas un LINESTRING, null est renvoyé.

Exemples

Le code SQL suivant renvoie une représentation de texte connu étendu (EWKT) d'un LINESTRING à quatre points à un objet GEOMETRY, et renvoie le point de départ de la linestring.

```
SELECT ST_AsEWKT(ST_StartPoint(ST_GeomFromText('LINESTRING(0 0,10 0,10 10,5 5,0
 5)',4326)));
```

```
st_asewkt
-------------
 SRID=4326;POINT(0 0)
```

ST_Touches

ST_Touches renvoie true si les projections 2D des deux géométries d'entrée se touchent. Les deux géométries se touchent si elles ne sont pas vides, se croisent et n'ont pas de points intérieurs communs.

Syntaxe

```
ST_Touches(geom1, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

BOOLEAN

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le code SQL suivant vérifie si un polygone touche une linestring.

```
SELECT ST_Touches(ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))'),
 ST_GeomFromText('LINESTRING(20 10,20 0,10 0)'));
```

```
 st_touches
-------------
 t
```

ST_Transform

ST_Transform renvoie une nouvelle géométrie dont les coordonnées sont transformées en système de référence spatiale défini par l'identifiant du système de référence spatiale (SRID) d'entrée.

Syntaxe

```
ST_Transform(geom, srid)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

srid

Valeur du type de données INTEGER représentant un SRID.

Type de retour

GEOMETRY.

La valeur de SRID de la géométrie renvoyée est définie sur srid.

Si geom ou srid est null, null est renvoyé.

Si la valeur SRID associée à l'entrée geom n'existe pas, une erreur est renvoyée.

Si srid n'existe pas, une erreur est renvoyée.

Exemples

Le code SQL suivant transforme le SRID d'une collection géométrique vide.

```
SELECT ST_AsEWKT(ST_Transform(ST_GeomFromText('GEOMETRYCOLLECTION EMPTY', 3857),
 4326));
```

```
            st_asewkt
-----------------------------------
 SRID=4326;GEOMETRYCOLLECTION EMPTY
```

Le code SQL suivant transforme le SRID d'une linestring.

```
SELECT ST_AsEWKT(ST_Transform(ST_GeomFromText('LINESTRING(110 40, 2 3, -10 80, -7 9,
 -22 -33)', 4326), 26918));
```

```
     st_asewkt
-----------------------------------------------------------------------------------------------
 SRID=26918;LINESTRING(73106.6977300955 15556182.9688576,14347201.5059964
 1545178.32934967,1515090.41262989 9522193.25115316,10491250.83295
 2575457.28410878,5672303.72135968 -5233682.61176205)
```

Le code SQL suivant transforme le SRID d'un polygone.

```
SELECT ST_AsEWKT(ST_Transform(ST_GeomFromText('POLYGON Z ((-10 10 -7, -65 10 -6, -10 64
 -5, -10 10 -7), (-11 11 5, -11 12 6, -12 11 7, -11 11 5))', 6989), 6317));
```

```
                                        st_asewkt
-----------------------------------------------------------------------------------------------
 SRID=6317;POLYGON Z ((6186430.2771091 -1090834.57212608
 1100247.33216237,2654831.67853801 -5693304.90741276 1100247.50581055,2760987.41750022
 -486836.575101877 5709710.44137268,6186430.2771091 -1090834.57212608
 1100247.33216237),(6146675.25029258 -1194792.63532103 1209007.1115113,6125027.87562215
 -1190584.81194058 1317403.77865723,6124888.99555252 -1301885.3455052
 1209007.49312929,6146675.25029258 -1194792.63532103 1209007.1115113))
```

ST_Union

ST_Union renvoie une géométrie représentant l'union de deux géométries. C'est-à-dire qu'il fusionne les géométries en entrée pour produire une géométrie résultante sans chevauchement.

Syntaxe

```
ST_Union(geom1, geom2)
```

Arguments

geom1

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

geom2

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

GEOMETRY

La valeur de l'identifiant de système de référence spatiale (SRID) de la géométrie renvoyée est la valeur SRID des géométries d'entrée.

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 ou geom2 est vide, une géométrie vide est renvoyée.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une linestring ou une multilinestring, une erreur est renvoyée.

Si geom1 ou geom2 n'est pas une géométrie bidimensionnelle (2D), une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie la géométrie non vide représentant l'intersection de deux géométries en entrée.

```
SELECT ST_AsEWKT(ST_Union(ST_GeomFromText('POLYGON((0 0,100 100,0 200,0 0))'),
 ST_GeomFromText('POLYGON((0 0,10 0,0 10,0 0))')));
```

```
        st_asewkt
-------------------------
 POLYGON((0 0,0 200,100 100,5 5,10 0,0 0))
```

ST_Within

ST_Within renvoie true si la projection 2D de la première géométrie en entrée se trouve dans la projection 2D de la deuxième géométrie en entrée.

Par exemple, la géométrie A se trouve dans la géométrie B si chaque point dans A est un point dans B, et si son intérieur comporte une intersection non vide.

ST_Within(A, B) équivaut à ST_Contains(B, A).

Syntaxe

```
ST_Within(geom1, geom2)
```

Arguments

geom1

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY. Cette valeur est comparée à geom2 afin de déterminer si elle se trouve dans geom2.

geom2

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

BOOLEAN

Si geom1 ou geom2 est null, null est renvoyé.

Si geom1 et geom2 n'ont pas la même valeur pour l'identifiant de système de référence spatiale (SRID), une erreur est renvoyée.

Si geom1 ou geom2 est une collection géométrique, une erreur est renvoyée.

Exemples

Le SQL suivant vérifie si le premier polygone se trouve dans le deuxième polygone.

```
SELECT ST_Within(ST_GeomFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeomFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'));
```

```
st_within
-----------
 true
```

ST_X

ST_X renvoie la première coordonnée d'un point d'entrée.

Syntaxe

```
ST_X(point)
```

Arguments

point

Valeur POINT du type de données GEOMETRY.

Type de retour

Valeur DOUBLE PRECISION de la première coordonnée.

If point est null, null est renvoyé.

Si point est le point vide, null est renvoyé.

Si point n'est pas un POINT, une erreur est renvoyée.

Exemples

Le SQL suivant renvoie la première coordonnée d'un point.

```
SELECT ST_X(ST_Point(1,2));
```

```
st_x
-----------
 1.0
```

ST_ XMax

ST_ XMax renvoie la première coordonnée maximale d'une géométrie en entrée.

Syntaxe

```
ST_XMax(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

Valeur DOUBLE PRECISION de la première coordonnée maximum.

Si geom est vide, null est renvoyé.

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant renvoie la première coordonnée la plus importante d'une linestring.

```
SELECT ST_XMax(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_xmax
-----------
 77.42
```

ST_ XMin

ST_ XMin renvoie la première coordonnée minimale d'une géométrie en entrée.

Syntaxe

```
ST_XMin(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

Valeur DOUBLE PRECISION de la première coordonnée minimum.

Si geom est vide, null est renvoyé.

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant renvoie la première coordonnée la moins importante d'une linestring.

```
SELECT ST_XMin(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_xmin
-----------
 77.27
```

ST_Y

ST_X renvoie la deuxième coordonnée d'un point d'entrée.

Syntaxe

```
ST_Y(point)
```

Arguments

point

Valeur POINT du type de données GEOMETRY.

Type de retour

Valeur DOUBLE PRECISION de la deuxième coordonnée.

If point est null, null est renvoyé.

Si point est le point vide, null est renvoyé.

Si point n'est pas un POINT, une erreur est renvoyée.

Exemples

Le SQL suivant renvoie la deuxième coordonnée d’un point.

```
SELECT ST_Y(ST_Point(1,2));
```

```
st_y
-----------
 2.0
```

ST_ YMax

ST_ YMax renvoie la deuxième coordonnée maximale d'une géométrie en entrée.

Syntaxe

```
ST_YMax(geom)
```

Arguments

geom

Valeur de type de données `GEOMETRY` ou expression qui est évaluée sur un type `GEOMETRY`.

Type de retour

Valeur `DOUBLE PRECISION` de la deuxième coordonnée maximum.

Si geom est vide, null est renvoyé.

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant renvoie la deuxième coordonnée la plus importante d’une linestring.

```
SELECT ST_YMax(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_ymax
-----------
 29.31
```

ST_ YMin

ST_ YMin renvoie la deuxième coordonnée minimale d'une géométrie en entrée.

Syntaxe

```
ST_YMin(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

Valeur DOUBLE PRECISION de la deuxième coordonnée minimum.

Si geom est vide, null est renvoyé.

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant renvoie la deuxième coordonnée la moins importante d’une linestring.

```
SELECT ST_YMin(ST_GeomFromText('LINESTRING(77.29 29.07,77.42 29.26,77.27 29.31,77.29
 29.07)'));
```

```
st_ymin
-----------
 29.07
```

ST_Z

ST_Z renvoie la coordonnée z d’un point d’entrée.

Syntaxe

```
ST_Z(point)
```

Arguments

point

Valeur POINT du type de données GEOMETRY.

Type de retour

Valeur DOUBLE PRECISION de la coordonnée m.

If point est null, null est renvoyé.

Si point est un point 2D ou 3DM, null est renvoyé.

Si point est le point vide, null est renvoyé.

Si point n'est pas un POINT, une erreur est renvoyée.

Exemples

Le code SQL suivant renvoie la coordonnée z d'un point dans une géométrie 3DZ.

```
SELECT ST_Z(ST_GeomFromEWKT('POINT Z (1 2 3)'));
```

```
st_z
-----------
 3
```

Le SQL suivant renvoie la coordonnée z d'un point dans une géométrie 4D.

```
SELECT ST_Z(ST_GeomFromEWKT('POINT ZM (1 2 3 4)'));
```

```
st_z
-----------
 3
```

ST_ ZMax

ST_ ZMax renvoie la z coordonnée maximale d'une géométrie en entrée.

Syntaxe

```
ST_ZMax(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

Valeur DOUBLE PRECISION de la coordonnée z maximum.

Si geom est vide, null est renvoyé.

Si geom est null, null est renvoyé.

Si geom est une géométrie 2D ou 3DM, null est renvoyé.

Exemples

Le code SQL suivant renvoie la coordonnée z la plus importante d'une linestring dans une géométrie 3DZ.

```
SELECT ST_ZMax(ST_GeomFromEWKT('LINESTRING Z (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_zmax
-----------
  8
```

Le code SQL suivant renvoie la coordonnée z la plus importante d'une linestring dans une géométrie 4D.

```
SELECT ST_ZMax(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_zmax
-----------
  10
```

ST_ ZMin

ST_ ZMin renvoie la z coordonnée minimale d'une géométrie en entrée.

Syntaxe

```
ST_ZMin(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

Valeur DOUBLE PRECISION de la coordonnée z minimum.

Si geom est vide, null est renvoyé.

Si geom est null, null est renvoyé.

Si geom est une géométrie 2D ou 3DM, null est renvoyé.

Exemples

Le code SQL suivant renvoie la coordonnée z la moins importante d'une linestring dans une géométrie 3DZ.

```
SELECT ST_ZMin(ST_GeomFromEWKT('LINESTRING Z (0 1 2, 3 4 5, 6 7 8)'));
```

```
st_zmin
-----------
  2
```

Le code SQL suivant renvoie la coordonnée z la moins importante d'une linestring dans une géométrie 4D.

```
SELECT ST_ZMin(ST_GeomFromEWKT('LINESTRING ZM (0 1 2 3, 4 5 6 7, 8 9 10 11)'));
```

```
st_zmin
-----------
  2
```

Supports BBox

Supporte les BBox retours « true » si la géométrie d'entrée prend en charge le codage avec un cadre de délimitation précalculé. Pour plus d'informations sur la prise en charge des cadres de délimitation, consultez Cadre de délimitation.

Syntaxe

```
SupportsBBox(geom)
```

Arguments

geom

Valeur de type de données GEOMETRY ou expression qui est évaluée sur un type GEOMETRY.

Type de retour

BOOLEAN

Si geom est null, null est renvoyé.

Exemples

Le code SQL suivant renvoie true car la géométrie du point d'entrée prend en charge l'encodage avec un cadre de délimitation.

```
SELECT SupportsBBox(AddBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
supportsbbox
--------------
t
```

Le code SQL suivant renvoie false car la géométrie du point d'entrée ne prend pas en charge l'encodage avec un cadre de délimitation.

```
SELECT SupportsBBox(DropBBox(ST_GeomFromText('POLYGON((0 0,1 0,0 1,0 0))')));
```

```
supportsbbox
--------------
f
```

Fonctions de chaîne

Rubriques

- Opérateur (concaténation) ||
- Fonction ASCII
- Fonction BPCHARCMP
- Fonction BTRIM
- Fonction BTTEXT_PATTERN_CMP
- Fonction CHAR_LENGTH
- Fonction CHARACTER_LENGTH
- Fonction CHARINDEX
- Fonction CHR
- Fonction COLLATE
- Fonction CONCAT
- CRC32 fonction
- Fonction DIFFERENCE
- Fonction INITCAP
- Fonctions LEFT et RIGHT
- Fonction LEN
- Fonction LENGTH

- Fonction LOWER
- Fonctions LPAD et RPAD
- Fonction LTRIM
- Fonction OCTETINDEX
- Fonction OCTET_LENGTH
- Fonction POSITION
- Fonction QUOTE_IDENT
- Fonction QUOTE_LITERAL
- Fonction REGEXP_COUNT
- Fonction REGEXP_INSTR
- Fonction REGEXP_REPLACE
- Fonction REGEXP_SUBSTR
- Fonction REPEAT
- Fonction REPLACE
- Fonction REPLICATE
- Fonction REVERSE
- Fonction RTRIM
- Fonction SOUNDEX
- Fonction SPLIT_PART
- Fonction STRPOS
- Fonction STRTOL
- Fonction SUBSTRING
- Fonction TEXTLEN
- Fonction TRANSLATE
- Fonction TRIM
- Fonction UPPER

Fonctions de chaîne qui traitent et manipulent des chaînes de caractères ou des expressions qui correspondent à des chaînes de caractères. Lorsque l'argument string de ces fonctions est une valeur littérale, il doit être entre guillemets simples. Les types de données pris en charge sont CHAR et VARCHAR.

La section suivante fournit les noms de fonctions, la syntaxe et les descriptions des fonctions prises en charge. Tous les décalages en chaînes sont basés sur un.

Fonctions de nœud principal uniquement obsolètes

Les fonctions de chaîne suivantes sont obsolètes, car elles s'exécutent uniquement sur le nœud principal. Pour plus d'informations, consultez Fonctions exécutées uniquement sur le nœud principal

- GET_BYTE
- SET_BIT
- SET_BYTE
- TO_ASCII

Opérateur (concaténation) ||

Concatène deux expressions de chaque côté du symbole `||` et renvoie l'expression concaténée.

Similaire à Fonction CONCAT.

> **Note**
>
> Si l'une des expressions (ou les deux) a la valeur null, le résultat de la concaténation est NULL.

Syntaxe

```
expression1 || expression2
```

Arguments

expression1

Chaîne CHAR, chaîne VARCHAR, expression binaire ou expression évaluée à l'un de ces types.

expression2

Chaîne CHAR, chaîne VARCHAR, expression binaire ou expression évaluée à l'un de ces types.

Type de retour

Le type de retour de la chaîne est le même que celui des arguments d'entrée. Par exemple, la concaténation de deux chaînes de type VARCHAR renvoie une chaîne de type VARCHAR.

Exemples

Les exemples suivants utilisent les tables USERS et VENUE de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour concaténer les champs FIRSTNAME et LASTNAME de la table USERS de l'exemple de base de données, utilisez l'exemple suivant.

```
SELECT (firstname || ' ' || lastname) as fullname
FROM users
ORDER BY 1
LIMIT 10;

+-----------------+
|    fullname     |
+-----------------+
| Aaron Banks     |
| Aaron Booth     |
| Aaron Browning  |
| Aaron Burnett   |
| Aaron Casey     |
| Aaron Cash      |
| Aaron Castro    |
| Aaron Dickerson |
| Aaron Dixon     |
| Aaron Dotson    |
+-----------------+
```

Pour concaténer des colonnes susceptibles de contenir des valeurs nulles, utilisez l'expression Fonctions NVL et COALESCE. L'exemple suivant utilise NVL pour renvoyer `0` chaque fois que la valeur NULL est rencontrée.

```
SELECT (venuename || ' seats ' || NVL(venueseats, 0)) as seating
FROM venue
WHERE venuestate = 'NV' or venuestate = 'NC'
ORDER BY 1
LIMIT 10;
```

```
+-------------------------------------+
|               seating               |
+-------------------------------------+
| Ballys Hotel seats 0                |
| Bank of America Stadium seats 73298 |
| Bellagio Hotel seats 0              |
| Caesars Palace seats 0              |
| Harrahs Hotel seats 0               |
| Hilton Hotel seats 0                |
| Luxor Hotel seats 0                 |
| Mandalay Bay Hotel seats 0          |
| Mirage Hotel seats 0                |
| New York New York seats 0           |
+-------------------------------------+
```

Fonction ASCII

La fonction ASCII renvoie le code ASCII, ou le point de code Unicode, du premier caractère de la chaîne que vous spécifiez. La fonction renvoie `0` si la chaîne est vide. Elle renvoie `NULL` si la chaîne a la valeur null.

Syntaxe

```
ASCII('string')
```

Argument

string

Chaîne `CHAR` ou chaîne `VARCHAR`.

Type de retour

INTEGER

Exemples

Pour renvoyer `NULL`, utilisez l'exemple suivant. La fonction NULLIF renvoie `NULL` si les deux arguments sont identiques, si bien que l'argument d'entrée de la fonction ASCII est `NULL`. Pour de plus amples informations, veuillez consulter Fonction NULLIF.

```
SELECT ASCII(NULLIF('',''));
```

```
+-------+
| ascii |
+-------+
|  NULL |
+-------+
```

Pour renvoyer le code ASCII 0, utilisez l'exemple suivant.

```
SELECT ASCII('');

+-------+
| ascii |
+-------+
|     0 |
+-------+
```

Pour renvoyer le code ASCII 97 pour la première lettre du mot amazon, utilisez l'exemple suivant.

```
SELECT ASCII('amazon');

+-------+
| ascii |
+-------+
|    97 |
+-------+
```

Pour renvoyer le code ASCII 65 pour la première lettre du mot Amazon, utilisez l'exemple suivant.

```
SELECT ASCII('Amazon');

+-------+
| ascii |
+-------+
|    65 |
+-------+
```

Fonction BPCHARCMP

Compare la valeur de deux chaînes et renvoie un nombre entier. Si les chaînes sont identiques, la fonction renvoie `0`. Si la première chaîne est supérieure dans l'ordre alphabétique, la fonction renvoie 1. Si la seconde chaîne est supérieure, la fonction renvoie `-1`.

Pour les caractères à plusieurs octets, la comparaison est basée sur l'encodage en octets.

Synonyme de Fonction BTTEXT_PATTERN_CMP.

Syntaxe

```
BPCHARCMP(string1, string2)
```

Arguments

string1

Chaîne CHAR ou chaîne VARCHAR.

string2

Chaîne CHAR ou chaîne VARCHAR.

Type de retour

INTEGER

Exemples

Les exemples suivants utilisent la table USERS de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour déterminer si le prénom d'un utilisateur est alphabétiquement supérieur au nom de famille de l'utilisateur pour les dix premières entrées de la table USERS, utilisez l'exemple suivant. Pour les entrées où la chaîne FIRSTNAME est située plus loin dans l'ordre alphabétique que la chaîne LASTNAME, la fonction renvoie 1. Si LASTNAME est situé plus loin dans l'ordre alphabétique que FIRSTNAME, la fonction renvoie -1.

```
SELECT userid, firstname, lastname, BPCHARCMP(firstname, lastname)
FROM users
ORDER BY 1, 2, 3, 4
LIMIT 10;

+--------+-----------+-----------+-----------+
| userid | firstname | lastname  | bpcharcmp |
+--------+-----------+-----------+-----------+
|      1 | Rafael    | Taylor    |        -1 |
```

```
|      2 | Vladimir  | Humphrey  |         1 |
|      3 | Lars      | Ratliff   |        -1 |
|      4 | Barry     | Roy       |        -1 |
|      5 | Reagan    | Hodge     |         1 |
|      6 | Victor    | Hernandez |         1 |
|      7 | Tamekah   | Juarez    |         1 |
|      8 | Colton    | Roy       |        -1 |
|      9 | Mufutau   | Watkins   |        -1 |
|     10 | Naida     | Calderon  |         1 |
+--------+-----------+-----------+-----------+
```

Pour renvoyer toutes les entrées de la table USERS si la fonction renvoie `0`, utilisez l'exemple suivant. La fonction renvoie `0` lorsque FIRSTNAME est identique à LASTNAME.

```
SELECT userid, firstname, lastname,
BPCHARCMP(firstname, lastname)
FROM users
WHERE BPCHARCMP(firstname, lastname)=0
ORDER BY 1, 2, 3, 4;

+--------+-----------+----------+-----------+
| userid | firstname | lastname | bpcharcmp |
+--------+-----------+----------+-----------+
|     62 | Chase     | Chase    |         0 |
|   4008 | Whitney   | Whitney  |         0 |
|  12516 | Graham    | Graham   |         0 |
|  13570 | Harper    | Harper   |         0 |
|  16712 | Cooper    | Cooper   |         0 |
|  18359 | Chase     | Chase    |         0 |
|  27530 | Bradley   | Bradley  |         0 |
|  31204 | Harding   | Harding  |         0 |
+--------+-----------+----------+-----------+
```

Fonction BTRIM

La fonction BTRIM tronque une chaîne en supprimant les espaces de début et de fin ou en supprimant les caractères de début et de fin qui correspondent à une chaîne spécifiée de manière facultative.

Syntaxe

```
BTRIM(string [, trim_chars ] )
```

Arguments

string

Chaîne VARCHAR d'entrée à tronquer.

trim_chars

Chaîne VARCHAR contenant les caractères à mettre en correspondance.

Type de retour

La fonction BTRIM renvoie une chaîne VARCHAR.

Exemples

L'exemple suivant tronque les espaces de début et de fin de la chaîne `' abc '` :

```
select '     abc    ' as untrim, btrim('     abc    ') as trim;

untrim    | trim
----------+------
   abc    | abc
```

L'exemple suivant supprime les chaînes `'xyz'` de début et de fin de la chaîne `'xyzaxyzbxyzcxyz'`. Les occurrences de début et de fin de `'xyz'` sont supprimées, mais celles qui se trouvent à l'intérieur de la chaîne sont conservées.

```
select 'xyzaxyzbxyzcxyz' as untrim,
btrim('xyzaxyzbxyzcxyz', 'xyz') as trim;

     untrim      |   trim
-----------------+-----------
 xyzaxyzbxyzcxyz | axyzbxyzc
```

L'exemple suivant supprime les parties de début et de fin de la chaîne `'setuphistorycassettes'` qui correspondent à l'un des caractères de la liste trim_chars `'tes'`. Tout caractère `t`, `e` ou `s` précédant un autre caractère qui ne figure pas dans la liste trim_chars au début ou à la fin de la chaîne d'entrée est supprimé.

```
SELECT btrim('setuphistorycassettes', 'tes');
```

```
     btrim
-----------------
 uphistoryca
```

Fonction BTTEXT_PATTERN_CMP

Synonyme de la fonction BPCHARCMP.

Consultez Fonction BPCHARCMP pour plus de détails.

Fonction CHAR_LENGTH

Synonyme de la fonction LEN.

Consultez Fonction LEN.

Fonction CHARACTER_LENGTH

Synonyme de la fonction LEN.

Consultez Fonction LEN.

Fonction CHARINDEX

Renvoie l'emplacement de la sous-chaîne spécifiée dans une chaîne.

Consultez Fonction POSITION et Fonction STRPOS pour des fonctions similaires.

Syntaxe

```
CHARINDEX( substring, string )
```

Arguments

substring

Sous-chaîne à rechercher dans la chaîne.

string

Chaîne ou colonne à rechercher.

Type de retour

INTEGER

La fonction CHARINDEX renvoie un `INTEGER` correspondant à la position de la sous-chaîne (de base 1, pas de base 0). La position est basée sur le nombre de caractères, pas d'octets, de sorte que les caractères à plusieurs octets soient comptés comme des caractères seuls. CHARINDEX renvoie `0` si la sous-chaîne ne figure pas dans la chaîne.

Exemples

Pour renvoyer la position de la chaîne `fish` dans le mot `dog`, utilisez l'exemple suivant.

```
SELECT CHARINDEX('fish', 'dog');

+-----------+
| charindex |
+-----------+
|         0 |
+-----------+
```

Pour renvoyer la position de la chaîne `fish` dans le mot `dogfish`, utilisez l'exemple suivant.

```
SELECT CHARINDEX('fish', 'dogfish');

+-----------+
| charindex |
+-----------+
|         4 |
+-----------+
```

L'exemple suivant utilise la table SALES de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour renvoyer le nombre de transactions de vente distinctes avec une commission supérieure à 999,00 dans la table SALES, utilisez l'exemple suivant. Cette commande compte les commissions supérieures à 999,00 en vérifiant si la décimale est à plus de 4 positions du début de la valeur de la commission.

```
SELECT DISTINCT CHARINDEX('.', commission), COUNT (CHARINDEX('.', commission))
FROM sales
```

```
WHERE CHARINDEX('.', commission) > 4
GROUP BY CHARINDEX('.', commission)
ORDER BY 1,2;

+-----------+-------+
| charindex | count |
+-----------+-------+
|         5 |   629 |
+-----------+-------+
```

Fonction CHR

La fonction CHR renvoie le caractère qui correspond à la valeur du point de code ASCII spécifiée par le paramètre d'entrée.

Syntaxe

```
CHR(number)
```

Argument

number

Le paramètre d'entrée est un `INTEGER` qui représente une valeur de point de code ASCII.

Type de retour

CHAR

La fonction CHR renvoie une chaîne `CHAR` si un caractère ASCII correspond à la valeur d'entrée. Si le nombre en entrée n'a aucun équivalent ASCII, la fonction renvoie `NULL`.

Exemples

Pour renvoyer le caractère correspondant au point de code ASCII 0, utilisez l'exemple suivant. Notez que la fonction CHR renvoie `NULL` pour l'entrée `0`.

```
SELECT CHR(0);

+-----+
| chr |
```

```
+-----+
|     |
+-----+
```

Pour renvoyer le caractère correspondant au point de code ASCII 65, utilisez l'exemple suivant.

```
SELECT CHR(65);

+-----+
| chr |
+-----+
| A   |
+-----+
```

Pour renvoyer les noms d'événements distincts qui commencent par un A majuscule (point de code ASCII 65), utilisez l'exemple suivant. L'exemple suivant utilise la table EVENT de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

```
SELECT DISTINCT eventname FROM event
WHERE SUBSTRING(eventname, 1, 1)=CHR(65) LIMIT 5;

+-----------------------+
|       eventname       |
+-----------------------+
| A Catered Affair      |
| As You Like It        |
| A Man For All Seasons |
| Alan Jackson          |
| Armando Manzanero     |
+-----------------------+
```

Fonction COLLATE

La fonction COLLATE remplace le classement d'une colonne ou d'une expression de chaîne.

Pour plus d'informations sur la création de tables à l'aide du classement de bases de données, consultez CREATE TABLE.

Pour plus d'informations sur la création de bases de données à l'aide du classement de bases de données, consultez CREATE DATABASE.

Syntaxe

```
COLLATE( string, 'case_sensitive' | 'case_insensitive');
```

Arguments

string

Colonne de chaîne ou expression que vous voulez remplacer.

'case_sensitive' | 'case_insensitive'

Constante de chaîne d'un nom de classement. Amazon Redshift ne prend en charge que case_sensitive ou case_insensitive.

Type de retour

La fonction COLLATE renvoie VARCHAR ou CHAR en fonction du premier type d'expression en entrée. Cette fonction ne modifie que le classement du premier argument d'entrée et ne modifie pas sa valeur de sortie.

Exemples

Pour créer la table T et définir col1 dans la table T comme `case_sensitive`, utilisez l'exemple suivant.

```
CREATE TABLE T ( col1 Varchar(20) COLLATE case_sensitive );

INSERT INTO T VALUES ('john'),('JOHN');
```

Lorsque vous exécutez la première requête, Amazon Redshift renvoie uniquement `john`. Après l'exécution de la fonction COLLATE sur col1, le classement devient `case_insensitive`. La deuxième requête renvoie à la fois `john` et `JOHN`.

```
SELECT * FROM T WHERE col1 = 'john';

+------+
| col1 |
+------+
| john |
```

```
+------+

SELECT * FROM T WHERE COLLATE(col1, 'case_insensitive') = 'john';

+------+
| col1 |
+------+
| john |
| JOHN |
+------+
```

Pour créer la table A et définir col1 dans la table A comme `case_insensitive`, utilisez l'exemple suivant.

```
CREATE TABLE A ( col1 Varchar(20) COLLATE case_insensitive );

INSERT INTO A VALUES ('john'),('JOHN');
```

Lorsque vous exécutez la première requête, Amazon Redshift renvoie à la fois `john` et `JOHN`. Après l'exécution de la fonction COLLATE sur col1, le classement devient `case_sensitive`. La deuxième requête renvoie uniquement `john`.

```
SELECT * FROM A WHERE col1 = 'john';

+------+
| col1 |
+------+
| john |
| JOHN |
+------+

SELECT * FROM A WHERE COLLATE(col1, 'case_sensitive') = 'john';

+------+
| col1 |
+------+
| john |
+------+
```

Fonction CONCAT

La fonction CONCAT concatène deux expressions et renvoie l'expression résultante. Pour concaténer plus de deux expressions, utilisez les fonction CONCAT imbriquées. L'opérateur de concaténation (| |) entre deux expressions donne les mêmes résultats que la fonction CONCAT.

Syntaxe

```
CONCAT ( expression1, expression2 )
```

Arguments

expression1, expression2

Les deux arguments peuvent être une chaîne de caractères de longueur fixe, une chaîne de caractères de longueur variable, une expression binaire ou une expression qui a pour résultat l'une de ces entrées.

Type de retour

CONCAT renvoie une expression. Le type de données de l'expression est le même que celui des arguments d'entrée.

Si les expressions d'entrée sont de types différents, Amazon Redshift essaie de « convertir » implicitement l'une des expressions. Si des valeurs ne peuvent pas être converties, une erreur est renvoyée.

Notes d'utilisation

- Pour la fonction CONCAT et l'opérateur de concaténation, si une expression ou les deux ont la valeur null, le résultat de la concaténation est null.

Exemples

L'exemple suivant concatène deux littéraux caractères :

```
SELECT CONCAT('December 25, ', '2008');

concat
-------------------
```

```
December 25, 2008
(1 row)
```

La requête suivante, utilisant l'opérateur | | au lieu de CONCAT, produit le même résultat :

```
SELECT 'December 25, '||'2008';

?column?
-------------------
December 25, 2008
(1 row)
```

L'exemple suivant utilise une fonction CONCAT imbriquée dans une autre fonction CONCAT pour concaténer trois chaînes de caractères :

```
SELECT CONCAT('Thursday, ', CONCAT('December 25, ', '2008'));

concat
----------------------------
Thursday, December 25, 2008
(1 row)
```

Pour concaténer des colonnes susceptibles de contenir NULLs, utilisez leFonctions NVL et COALESCE, qui renvoie une valeur donnée lorsqu'il rencontre la valeur NULL. L'exemple suivant utilise NVL pour renvoyer un 0 chaque fois que la valeur NULL est rencontrée.

```
SELECT CONCAT(venuename, CONCAT(' seats ', NVL(venueseats, 0))) AS seating
FROM venue WHERE venuestate = 'NV' OR venuestate = 'NC'
ORDER BY 1
LIMIT 5;

seating
-----------------------------------
Ballys Hotel seats 0
Bank of America Stadium seats 73298
Bellagio Hotel seats 0
Caesars Palace seats 0
Harrahs Hotel seats 0
(5 rows)
```

La requête suivante concatène les valeurs CITY et STATE de la table VENUE :

```
SELECT CONCAT(venuecity, venuestate)
FROM venue
WHERE venueseats > 75000
ORDER BY venueseats;

concat
-------------------
DenverCO
Kansas CityMO
East RutherfordNJ
LandoverMD
(4 rows)
```

La requête suivante utilise des fonctions CONCAT imbriquées. La requête concatène les valeurs CITY et STATE de la table VENUE, mais délimite la chaîne qui en résulte par une virgule et un espace :

```
SELECT CONCAT(CONCAT(venuecity,', '),venuestate)
FROM venue
WHERE venueseats > 75000
ORDER BY venueseats;

concat
---------------------
Denver, CO
Kansas City, MO
East Rutherford, NJ
Landover, MD
(4 rows)
```

L'exemple suivant concatène deux expressions binaires. Où `abc` est une valeur binaire (avec une représentation hexadécimale de `616263`) et `def` est une valeur binaire (avec une représentation hexadécimale de `646566`). Le résultat est automatiquement affiché sous forme de représentation hexadécimale de la valeur binaire.

```
SELECT CONCAT('abc'::VARBYTE, 'def'::VARBYTE);

concat
-------------------
616263646566
```

CRC32 fonction

CRC32 est une fonction utilisée pour détecter les erreurs. La fonction utilise un CRC32 algorithme pour détecter les changements entre les données source et cible. La CRC32 fonction convertit une chaîne de longueur variable en une chaîne de 8 caractères qui est une représentation textuelle de la valeur hexadécimale d'une séquence binaire de 32 bits. Pour détecter les modifications entre les données source et cible, utilisez la CRC32 fonction sur les données source et stockez la sortie. Utilisez ensuite la CRC32 fonction sur les données cibles et comparez cette sortie à la sortie des données source. Les sorties seront les mêmes si les données n'ont pas été modifiées, et les sorties seront différentes si les données ont été modifiées.

Syntaxe

```
CRC32(string)
```

Arguments

string

Chaîne CHAR, chaîne VARCHAR ou expression qui équivaut implicitement à un type CHAR ou VARCHAR.

Type de retour

La CRC32 fonction renvoie une chaîne de 8 caractères qui est une représentation textuelle de la valeur hexadécimale d'une séquence binaire de 32 bits. La CRC32 fonction Amazon Redshift est basée sur le polynôme CRC-32C.

Exemples

Pour montrer la valeur de 8 bits de la chaîne `Amazon Redshift`.

```
SELECT CRC32('Amazon Redshift');

+----------+
|  crc32   |
+----------+
| f2726906 |
+----------+
```

Fonction DIFFERENCE

La fonction DIFFERENCE compare les codes Soundex américains de deux chaînes. La fonction renvoie un `INTEGER` pour indiquer le nombre de caractères correspondants entre les codes Soundex.

Un code Soundex est une chaîne de quatre caractères. Un code Soundex représente la sonorité d'un mot plutôt que son orthographe. Par exemple, `Smith` et `Smyth` ont le même code Soundex.

Syntaxe

```
DIFFERENCE(string1, string2)
```

Arguments

string1

Chaîne `CHAR`, chaîne `VARCHAR` ou expression qui équivaut implicitement à un type `CHAR` ou `VARCHAR`.

string2

Chaîne `CHAR`, chaîne `VARCHAR` ou expression qui équivaut implicitement à un type `CHAR` ou `VARCHAR`.

Type de retour

INTEGER

La fonction DIFFERENCE renvoie une valeur `INTEGER` comprise entre 0 et 4 qui compte le nombre de caractères correspondants dans les codes Soundex américains des deux chaînes. Un code Soundex comporte 4 caractères, donc la fonction DIFFERENCE renvoie `4` quand les 4 caractères des valeurs de code Soundex américain des chaînes sont identiques. DIFFERENCE renvoie `0` si l'une des deux chaînes est vide. La fonction renvoie 1 si aucune des chaînes ne contient de caractères valides. La fonction DIFFERENCE convertit uniquement les caractères ASCII alphabétiques anglais minuscules ou majuscules, y compris a–z et A–Z. DIFFERENCE ignore les autres caractères.

Exemples

Pour comparer les valeurs Soundex des chaînes `%` et `@`, utilisez l'exemple suivant. La fonction renvoie 1 car aucune des chaînes ne contient de caractères valides.

```
SELECT DIFFERENCE('%', '@');

+------------+
| difference |
+------------+
|          1 |
+------------+
```

Pour comparer les valeurs Soundex de `Amazon` et d'une chaîne vide, utilisez l'exemple suivant. La fonction renvoie `0` car l'une des deux chaînes est vide.

```
SELECT DIFFERENCE('Amazon', '');

+------------+
| difference |
+------------+
|          0 |
+------------+
```

Pour comparer les valeurs Soundex des chaînes `Amazon` et `Ama`, utilisez l'exemple suivant. La fonction renvoie 2 car 2 caractères des valeurs Soundex des chaînes sont identiques.

```
SELECT DIFFERENCE('Amazon', 'Ama');

+------------+
| difference |
+------------+
|          2 |
+------------+
```

Pour comparer les valeurs Soundex des chaînes `Amazon` et `+-*/%Amazon`, utilisez l'exemple suivant. La fonction renvoie `4` car les quatre caractères des valeurs Soundex des chaînes sont identiques. Notez que la fonction ignore les caractères non valides `+-*/%` dans la deuxième chaîne.

```
SELECT DIFFERENCE('Amazon', '+-*/%Amazon');
```

```
+------------+
| difference |
+------------+
|          4 |
+------------+
```

Pour comparer les valeurs Soundex des chaînes `AC/DC` et `Ay See Dee See`, utilisez l'exemple suivant. La fonction renvoie `4` car les quatre caractères des valeurs Soundex des chaînes sont identiques.

```
SELECT DIFFERENCE('AC/DC', 'Ay See Dee See');

+------------+
| difference |
+------------+
|          4 |
+------------+
```

Fonction INITCAP

Met en majuscules la première lettre de chaque mot d'une chaîne spécifiée. INITCAP prend en charge les caractères à plusieurs octets UTF-8, à concurrence de quatre octets au maximum par caractère.

Syntaxe

```
INITCAP(string)
```

Argument

string

Chaîne `CHAR`, chaîne `VARCHAR` ou expression qui équivaut implicitement à un type `CHAR` ou `VARCHAR`.

Type de retour

VARCHAR

Notes d'utilisation

La fonction INITCAP met la première lettre de chaque mot d'une chaîne en majuscules, et les lettres suivantes en minuscules (ou à gauche). Par conséquent, il est important de comprendre quels caractères (autres que l'espace) servent de séparateurs de mots. Un caractère séparateur de mots est tout caractère non alphanumérique, y compris des signes de ponctuation, des symboles et des caractères de contrôle. Tous les caractères suivants sont des séparateurs de mots :

```
! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [ \ ] ^ _ ` { | } ~
```

Tabulation, caractères de nouvelle de ligne, sauts de page, sauts de ligne et les retours à la ligne sont également des séparateurs de mots.

Exemples

Les exemples suivants utilisent les données des tables CATEGORY et USERS de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour mettre en majuscules les initiales de chaque mot dans la colonne CATDESC, utilisez l'exemple suivant.

```
SELECT catid, catdesc, INITCAP(catdesc)
FROM category
ORDER BY 1, 2, 3;

+-------+--------------------------------------------
+-----------------------------------------+
| catid |                  catdesc                   |                  initcap
          |
+-------+--------------------------------------------
+-----------------------------------------+
|     1 | Major League Baseball                      | Major League Baseball
          |
|     2 | National Hockey League                     | National Hockey League
          |
|     3 | National Football League                   | National Football League
          |
|     4 | National Basketball Association            | National Basketball Association
          |
|     5 | Major League Soccer                        | Major League Soccer
          |
```

```
|     6 | Musical theatre                              | Musical Theatre
          |
|     7 | All non-musical theatre                      | All Non-Musical Theatre
          |
|     8 | All opera and light opera                    | All Opera And Light Opera
          |
|     9 | All rock and pop music concerts              | All Rock And Pop Music Concerts
          |
|    10 | All jazz singers and bands                   | All Jazz Singers And Bands
          |
|    11 | All symphony, concerto, and choir concerts | All Symphony, Concerto, And
 Choir Concerts |
+-------+----------------------------------------------
+-----------------------------------------+
```

Pour montrer que la fonction INITCAP ne conserve pas les majuscules quand celles-ci ne figurent pas au début des mots, utilisez l'exemple suivant. Par exemple, la chaîne `MLB` devient `Mlb`.

```
SELECT INITCAP(catname)
FROM category
ORDER BY catname;

+-----------+
|  initcap  |
+-----------+
| Classical |
| Jazz      |
| Mlb       |
| Mls       |
| Musicals  |
| Nba       |
| Nfl       |
| Nhl       |
| Opera     |
| Plays     |
| Pop       |
+-----------+
```

Pour montrer que les caractères non alphanumériques autres que les espaces servent de séparateurs de mots, utilisez l'exemple suivant. Plusieurs lettres de chaque chaîne seront mises en majuscules.

```
SELECT email, INITCAP(email)
FROM users
ORDER BY userid DESC LIMIT 5;

+-----------------------------------+-----------------------------------+
|               email               |              initcap              |
+-----------------------------------+-----------------------------------+
| urna.Ut@egetdictumplacerat.edu    | Urna.Ut@Egetdictumplacerat.Edu    |
| nibh.enim@egestas.ca              | Nibh.Enim@Egestas.Ca              |
| in@Donecat.ca                     | In@Donecat.Ca                     |
| sodales@blanditviverraDonec.ca    | Sodales@Blanditviverradonec.Ca    |
| sociis.natoque.penatibus@vitae.org | Sociis.Natoque.Penatibus@Vitae.Org |
+-----------------------------------+-----------------------------------+
```

Fonctions LEFT et RIGHT

Ces fonctions renvoient le nombre de caractères spécifié le plus à gauche ou le plus à droite dans une chaîne de caractères.

Le chiffre est basé sur le nombre de caractères, pas d'octets, de sorte que les caractères à plusieurs octets soient comptés comme des caractères seuls.

Syntaxe

```
LEFT( string,  integer )

RIGHT( string,  integer )
```

Arguments

string

Chaîne CHAR, chaîne VARCHAR ou expression qui équivaut à une chaîne CHAR ou VARCHAR.

integer

Nombre entier positif.

Type de retour

VARCHAR

Exemples

L'exemple suivant utilise les données de la table EVENT de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour renvoyer les 5 caractères les plus à gauche et les 5 caractères les plus à droite des noms d'événements dont l'événement est IDs compris entre 1 000 et 1 005, utilisez l'exemple suivant.

```
SELECT eventid, eventname,
LEFT(eventname,5) AS left_5,
RIGHT(eventname,5) AS right_5
FROM event
WHERE eventid BETWEEN 1000 AND 1005
ORDER BY 1;

+---------+----------------+--------+---------+
| eventid |   eventname    | left_5 | right_5 |
+---------+----------------+--------+---------+
|    1000 | Gypsy          | Gypsy  | Gypsy   |
|    1001 | Chicago        | Chica  | icago   |
|    1002 | The King and I | The K  | and I   |
|    1003 | Pal Joey       | Pal J  | Joey    |
|    1004 | Grease         | Greas  | rease   |
|    1005 | Chicago        | Chica  | icago   |
+---------+----------------+--------+---------+
```

Fonction LEN

Renvoie la longueur de la chaîne spécifiée en tant que nombre de caractères.

Syntaxe

LEN est synonyme de Fonction LENGTH, Fonction CHAR_LENGTH, Fonction CHARACTER_LENGTH, et Fonction TEXTLEN.

```
LEN(expression)
```

Argument

expression

Chaîne CHAR, chaîne VARCHAR, expression VARBYTE ou expression qui équivaut implicitement à un type CHAR, VARCHAR ou VARBYTE.

Type de retour

INTEGER

La fonction LEN renvoie un nombre entier indiquant le nombre de caractères dans la chaîne d'entrée.

Si la chaîne d'entrée est une chaîne de caractères, la fonction LEN renvoie le nombre de caractères dans les chaînes de plusieurs octets, pas le nombre d'octets. Par exemple, une colonne `VARCHAR(12)` est nécessaire pour stocker trois caractères chinois de quatre octets. La fonction LEN renvoie `3` pour cette même chaîne. Pour obtenir la longueur d'une chaîne en octets, utilisez la fonction OCTET_LENGTH.

Notes d'utilisation

Si expression est une chaîne `CHAR`, les espaces de fin ne sont pas comptés.

Si expression est une chaîne `VARCHAR`, les espaces de fin sont comptés.

Exemples

Pour renvoyer le nombre d'octets et le nombre de caractères dans la chaîne `français`, utilisez l'exemple suivant.

```
SELECT OCTET_LENGTH('français'),
LEN('français');

+--------------+-----+
| octet_length | len |
+--------------+-----+
|            9 |   8 |
+--------------+-----+
```

Pour renvoyer le nombre d'octets et le nombre de caractères dans la chaîne `français` sans utiliser la fonction OCTET_LENGTH, utilisez l'exemple suivant. Pour de plus amples informations, veuillez consulter Fonction CAST.

```
SELECT LEN(CAST('français' AS VARBYTE)) as bytes, LEN('français');

+-------+-----+
| bytes | len |
```

```
+-------+-----+
|     9 |   8 |
+-------+-----+
```

Pour renvoyer le nombre de caractères dans les chaînes `cat` sans les espaces de fin, `cat  ` avec trois espaces de fin, `cat  ` avec trois espaces de fin convertis en CHAR de longueur 6, et `cat  ` avec trois espaces de fin convertis en VARCHAR de longueur 6, utilisez l'exemple suivant. Notez que la fonction ne compte pas les espaces de fin des chaînes CHAR, mais qu'elle compte les espaces de fin des chaînes VARCHAR.

```
SELECT LEN('cat'), LEN('cat   '), LEN(CAST('cat   ' AS CHAR(6))) AS len_char,
 LEN(CAST('cat   ' AS VARCHAR(6))) AS len_varchar;

+-----+-----+----------+-------------+
| len | len | len_char | len_varchar |
+-----+-----+----------+-------------+
|   3 |   6 |        3 |           6 |
+-----+-----+----------+-------------+
```

L'exemple suivant utilise les données de la table VENUE de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour renvoyer les 10 noms de lieux les plus longs de la table VENUE, utilisez l'exemple suivant.

```
SELECT venuename, LEN(venuename)
FROM venue
ORDER BY 2 DESC, 1
LIMIT 10;

+----------------------------------------+-----+
|               venuename                | len |
+----------------------------------------+-----+
| Saratoga Springs Performing Arts Center |  39 |
| Lincoln Center for the Performing Arts  |  38 |
| Nassau Veterans Memorial Coliseum       |  33 |
| Jacksonville Municipal Stadium          |  30 |
| Rangers BallPark in Arlington           |  29 |
| University of Phoenix Stadium           |  29 |
| Circle in the Square Theatre            |  28 |
| Hubert H. Humphrey Metrodome            |  28 |
| Oriole Park at Camden Yards             |  27 |
| Dick's Sporting Goods Park              |  26 |
```

```
+----------------------------------------+-----+
```

Fonction LENGTH

Synonyme de la fonction LEN.

Consultez Fonction LEN.

Fonction LOWER

Convertit une chaîne en minuscules. LOWER prend en charge les caractères à plusieurs octets UTF-8, à concurrence de quatre octets au maximum par caractère.

Syntaxe

```
LOWER(string)
```

Argument

string

Chaîne VARCHAR ou expression qui équivaut au type VARCHAR.

Type de retour

chaîne

La fonction LOWER renvoie une chaîne qui est du même type que la chaîne en entrée. Par exemple, si l'entrée est une chaîne CHAR, la fonction renvoie une chaîne CHAR.

Exemples

L'exemple suivant utilise les données de la table CATEGORY de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour convertir les chaînes VARCHAR de la colonne CATNAME en minuscules, utilisez l'exemple suivant.

```
SELECT catname, LOWER(catname) FROM category ORDER BY 1,2;
```

```
+-----------+-----------+
|  catname  |   lower   |
+-----------+-----------+
| Classical | classical |
| Jazz      | jazz      |
| MLB       | mlb       |
| MLS       | mls       |
| Musicals  | musicals  |
| NBA       | nba       |
| NFL       | nfl       |
| NHL       | nhl       |
| Opera     | opera     |
| Plays     | plays     |
| Pop       | pop       |
+-----------+-----------+
```

Fonctions LPAD et RPAD

Ces fonctions ajoutent des caractères en préfixe ou en suffixe à une chaîne, en fonction d'une longueur spécifiée.

Syntaxe

```
LPAD(string1, length, [ string2 ])
```

```
RPAD(string1, length, [ string2 ])
```

Arguments

string1

Chaîne CHAR, chaîne VARCHAR ou expression qui équivaut implicitement à un type CHAR ou VARCHAR.

longueur

Nombre entier qui définit la longueur du résultat de la fonction. La longueur d'une chaîne est basée sur le nombre de caractères, pas d'octets, afin que les caractères à plusieurs octets soient comptés comme des caractères seuls. Si string1 dépasse la longueur spécifiée, il est tronqué (à droite). Si length est égal à zéro ou est un nombre négatif, le résultat de la fonction est une chaîne vide.

string2

(Facultatif) Un ou plusieurs caractères ajoutés en préfixe ou en suffixe à string1. Si cet argument n'est pas spécifié, les espaces sont utilisés.

Type de retour

VARCHAR

Exemples

Les exemples suivants utilisent les données de la table EVENT de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour tronquer un ensemble spécifié de noms d'événements à 20 caractères et ajouter en préfixe des espaces aux noms plus courts, utilisez l'exemple suivant.

```
SELECT LPAD(eventname, 20) FROM event
WHERE eventid BETWEEN 1 AND 5 ORDER BY 1;

+----------------------+
|         lpad         |
+----------------------+
|              Salome |
|        Il Trovatore |
|       Boris Godunov |
|     Gotterdammerung |
|La Cenerentola (Cind |
+----------------------+
```

Pour tronquer le même ensemble de noms d'événements à 20 caractères, mais ajouter `0123456789` comme suffixe aux noms plus courts, utilisez l'exemple suivant.

```
SELECT RPAD(eventname, 20,'0123456789') FROM event
WHERE eventid BETWEEN 1 AND 5 ORDER BY 1;

+----------------------+
|         rpad         |
+----------------------+
| Boris Godunov0123456 |
| Gotterdammerung01234 |
```

```
| Il Trovatore01234567 |
| La Cenerentola (Cind |
| Salome01234567890123 |
+----------------------+
```

Fonction LTRIM

Supprime les caractères du début d'une chaîne de caractères. Supprime la chaîne la plus longue ne contenant que des caractères de la liste des caractères supprimés. La suppression est terminée lorsqu'un caractère supprimé n'apparaît pas dans la chaîne d'entrée.

Syntaxe

```
LTRIM( string [, trim_chars] )
```

Arguments

string

Une colonne de chaîne, une expression ou un littéral de chaîne à supprimer.

trim_chars

Une colonne, une expression ou un littéral de chaîne qui représente les caractères à supprimer au début de la chaîne. Si la valeur n'est pas spécifiée, un espace est utilisé comme caractère de séparation.

Type de retour

La fonction LTRIM renvoie une chaîne de caractères qui est du même type que la chaîne d'entrée (CHAR ou VARCHAR).

Exemples

L'exemple suivant supprime l'année de la colonne `listime`. Les caractères supprimés dans la chaîne littérale `'2008-'` indiquent les caractères à supprimer à partir de la gauche. Si vous utilisez les caractères de suppression `'028-'`, vous obtiendrez le même résultat.

```
select listid, listtime, ltrim(listtime, '2008-')
from listing
order by 1, 2, 3
```

```
limit 10;

listid |      listtime       |     ltrim
-------+---------------------+---------------
     1 | 2008-01-24 06:43:29 | 1-24 06:43:29
     2 | 2008-03-05 12:25:29 | 3-05 12:25:29
     3 | 2008-11-01 07:35:33 | 11-01 07:35:33
     4 | 2008-05-24 01:18:37 | 5-24 01:18:37
     5 | 2008-05-17 02:29:11 | 5-17 02:29:11
     6 | 2008-08-15 02:08:13 | 15 02:08:13
     7 | 2008-11-15 09:38:15 | 11-15 09:38:15
     8 | 2008-11-09 05:07:30 | 11-09 05:07:30
     9 | 2008-09-09 08:03:36 | 9-09 08:03:36
    10 | 2008-06-17 09:44:54 | 6-17 09:44:54
```

LTRIM supprime les caractères de trim_chars lorsqu'ils apparaissent au début de la chaîne. L'exemple suivant supprime les caractères C, D et G lorsqu'ils figurent au début de VENUENAME, qui est une colonne VARCHAR.

```
select venueid, venuename, ltrim(venuename, 'CDG')
from venue
where venuename like '%Park'
order by 2
limit 7;

venueid | venuename                  | btrim
--------+----------------------------+--------------------------
    121 | ATT Park                   | ATT Park
    109 | Citizens Bank Park         | itizens Bank Park
    102 | Comerica Park              | omerica Park
      9 | Dick's Sporting Goods Park | ick's Sporting Goods Park
     97 | Fenway Park                | Fenway Park
    112 | Great American Ball Park   | reat American Ball Park
    114 | Miller Park                | Miller Park
```

L'exemple suivant utilise le caractère de suppression 2 qui est extrait de la colonne `venueid`.

```
select ltrim('2008-01-24 06:43:29', venueid)
from venue where venueid=2;

ltrim
```

```
------------------
008-01-24 06:43:29
```

L'exemple suivant ne supprime aucun caractère car 2 est trouvé avant le caractère de suppression `'0'`.

```
select ltrim('2008-01-24 06:43:29', '0');

ltrim
-------------------
2008-01-24 06:43:29
```

L'exemple suivant utilise le caractère de suppression d'espace par défaut et supprime les deux espaces du début de la chaîne.

```
select ltrim('  2008-01-24 06:43:29');

ltrim
-------------------
2008-01-24 06:43:29
```

Fonction OCTETINDEX

La fonction OCTETINDEX renvoie l'emplacement d'une sous-chaîne dans une chaîne sous la forme d'un nombre d'octets.

Syntaxe

```
OCTETINDEX(substring, string)
```

Arguments

substring

Chaîne CHAR, chaîne VARCHAR ou expression qui équivaut implicitement à un type CHAR ou VARCHAR.

string

Chaîne CHAR, chaîne VARCHAR ou expression qui équivaut implicitement à un type CHAR ou VARCHAR.

Type de retour

INTEGER

La fonction OCTETINDEX renvoie une valeur `INTEGER` correspondant à la position de substring dans string sous la forme d'un nombre d'octets, où le premier caractère de string est compté comme 1. Si string ne contient pas de caractères multioctets, le résultat est égal au résultat de la fonction CHARINDEX. Si string ne contient pas substring, la fonction renvoie `0`. Si substring est vide, la fonction renvoie 1.

Exemples

Pour renvoyer la position de la sous-chaîne q dans la chaîne `Amazon Redshift`, utilisez l'exemple suivant. Cet exemple renvoie `0` parce que substring ne figure pas dans string.

```
SELECT OCTETINDEX('q', 'Amazon Redshift');

+------------+
| octetindex |
+------------+
|          0 |
+------------+
```

Pour renvoyer la position d'une sous-chaîne vide dans la chaîne `Amazon Redshift`, utilisez l'exemple suivant. Cet exemple renvoie 1 parce que substring est vide.

```
SELECT OCTETINDEX('', 'Amazon Redshift');

+------------+
| octetindex |
+------------+
|          1 |
+------------+
```

Pour renvoyer la position de la sous-chaîne `Redshift` dans la chaîne `Amazon Redshift`, utilisez l'exemple suivant. Cet exemple renvoie 8 car substring commence au huitième octet de string.

```
SELECT OCTETINDEX('Redshift', 'Amazon Redshift');

+------------+
```

```
| octetindex |
+------------+
|          8 |
+------------+
```

Pour renvoyer la position de la sous-chaîne `Redshift` dans la chaîne `Amazon Redshift`, utilisez l'exemple suivant. Cet exemple renvoie 21 car les six premiers caractères de string sont des caractères à deux octets.

```
SELECT OCTETINDEX('Redshift', 'Άμαζον Amazon Redshift');

+------------+
| octetindex |
+------------+
|         21 |
+------------+
```

Fonction OCTET_LENGTH

Renvoie la longueur de la chaîne spécifiée en tant que nombre d'octets.

Syntaxe

```
OCTET_LENGTH(expression)
```

Argument

expression

Chaîne `CHAR`, chaîne `VARCHAR`, expression `VARBYTE` ou expression qui équivaut implicitement à un type `CHAR`, `VARCHAR` ou `VARBYTE`.

Type de retour

INTEGER

La fonction OCTET_LENGTH renvoie un nombre entier indiquant le nombre d'octets dans la chaîne d'entrée.

Si la chaîne d'entrée est une chaîne de caractères, la fonction LEN renvoie le nombre de caractères dans des chaînes de plusieurs octets, pas le nombre d'octets. Par exemple, une

colonne VARCHAR(12) est nécessaire pour stocker trois caractères chinois de quatre octets. La fonction OCTET_LENGTH renvoie 12 pour cette chaîne et la fonction LEN renvoie 3 pour cette même chaîne.

Notes d'utilisation

Si expression est une chaîne CHAR, la fonction renvoie la longueur de la chaîne CHAR. Par exemple, la sortie d'une entrée CHAR(6) est un CHAR(6).

Si expression est une chaîne VARCHAR, les espaces de fin sont comptés.

Exemples

Pour renvoyer le nombre d'octets lorsque la chaîne francais avec trois espaces de fin est convertie en type CHAR et VARCHAR, utilisez l'exemple suivant. Pour de plus amples informations, veuillez consulter Fonction CAST.

```
SELECT OCTET_LENGTH(CAST('francais   ' AS CHAR(15))) AS octet_length_char,
 OCTET_LENGTH(CAST('francais   ' AS VARCHAR(15))) AS octet_length_varchar;

+------------------+----------------------+
| octet_length_char | octet_length_varchar |
+------------------+----------------------+
|                15 |                   11 |
+------------------+----------------------+
```

Pour renvoyer le nombre d'octets et le nombre de caractères dans la chaîne français, utilisez l'exemple suivant.

```
SELECT OCTET_LENGTH('français'), LEN('français');

+--------------+-----+
| octet_length | len |
+--------------+-----+
|            9 |   8 |
+--------------+-----+
```

Pour renvoyer le nombre d'octets lorsque la chaîne français est convertie en VARBYTE, utilisez l'exemple suivant.

```
SELECT OCTET_LENGTH(CAST('français' AS VARBYTE));
```

```
+--------------+
| octet_length |
+--------------+
|            9 |
+--------------+
```

Fonction POSITION

Renvoie l'emplacement de la sous-chaîne spécifiée dans une chaîne.

Consultez Fonction CHARINDEX et Fonction STRPOS pour des fonctions similaires.

Syntaxe

```
POSITION(substring IN string )
```

Arguments

substring

Sous-chaîne à rechercher dans la chaîne.

string

Chaîne ou colonne à rechercher.

Type de retour

La fonction POSITION renvoie un `INTEGER` correspondant à la position de la sous-chaîne (de base 1, pas de base 0). La position est basée sur le nombre de caractères, pas d'octets, de sorte que les caractères à plusieurs octets soient comptés comme des caractères seuls. POSITION renvoie `0` si la sous-chaîne ne figure pas dans la chaîne.

Exemples

Pour renvoyer la position de la chaîne `fish` dans le mot `dog`, utilisez l'exemple suivant.

```
SELECT POSITION('fish' IN 'dog');

+-----------+
```

```
|  position |
+-----------+
|         0 |
+-----------+
```

Pour renvoyer la position de la chaîne `fish` dans le mot `dogfish`, utilisez l'exemple suivant.

```
SELECT POSITION('fish' IN 'dogfish');

+-----------+
|  position |
+-----------+
|         4 |
+-----------+
```

L'exemple suivant utilise la table SALES de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour renvoyer le nombre de transactions de vente distinctes avec une commission supérieure à 999,00 dans la table SALES, utilisez l'exemple suivant. Cette commande compte les commissions supérieures à 999,00 en vérifiant si la décimale est à plus de 4 positions du début de la valeur de la commission.

```
SELECT DISTINCT POSITION('.' IN commission), COUNT (POSITION('.' IN commission))
FROM sales
WHERE POSITION('.' IN commission) > 4
GROUP BY POSITION('.' IN commission)
ORDER BY 1,2;

+-----------+-------+
|  position | count |
+-----------+-------+
|         5 |   629 |
+-----------+-------+
```

Fonction QUOTE_IDENT

La fonction QUOTE_IDENT renvoie la chaîne spécifiée sous forme de chaîne entourée de guillemets doubles. La sortie de la fonction peut être utilisée comme identificateur dans une instruction SQL. La fonction double de manière appropriée tous les guillemets doubles intégrés.

QUOTE_IDENT ajoute des guillemets doubles uniquement lorsque cela est nécessaire pour créer un identifiant valide, lorsque la chaîne contient des caractères ne faisant pas partie d'un identifiant ou lorsqu'elle serait autrement convertie en minuscules. Pour renvoyer systématiquement une chaîne avec des guillemets simples, utilisez QUOTE_LITERAL.

Syntaxe

```
QUOTE_IDENT(string)
```

Argument

string

Chaîne CHAR ou VARCHAR.

Type de retour

La fonction QUOTE_IDENT renvoie le même type de chaîne que celui de l'argument string en entrée.

Exemples

Pour renvoyer la chaîne "CAT" avec des guillemets doublés, utilisez l'exemple suivant.

```
SELECT QUOTE_IDENT('"CAT"');

+-------------+
| quote_ident |
+-------------+
| """CAT"""   |
+-------------+
```

L'exemple suivant utilise les données de la table CATEGORY de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour renvoyer la colonne CATNAME entourée de guillemets, utilisez l'exemple suivant.

```
SELECT catid, QUOTE_IDENT(catname)
FROM category
ORDER BY 1,2;
```

```
+-------+-------------+
| catid | quote_ident |
+-------+-------------+
|     1 | "MLB"       |
|     2 | "NHL"       |
|     3 | "NFL"       |
|     4 | "NBA"       |
|     5 | "MLS"       |
|     6 | "Musicals"  |
|     7 | "Plays"     |
|     8 | "Opera"     |
|     9 | "Pop"       |
|    10 | "Jazz"      |
|    11 | "Classical" |
+-------+-------------+
```

Fonction QUOTE_LITERAL

La fonction QUOTE_LITERAL renvoie la chaîne spécifiée sous forme de chaîne entre guillemets simples pour qu'elle puisse être utilisée comme littéral de chaîne dans une instruction SQL. Si le paramètre d'entrée est un nombre, QUOTE_LITERAL le traite comme une chaîne. Double de manière appropriée les barres obliques inverses et les guillemets simples imbriqués.

Syntaxe

```
QUOTE_LITERAL(string)
```

Argument

string

Chaîne CHAR ou VARCHAR.

Type de retour

La fonction QUOTE_LITERAL renvoie une chaîne CHAR ou VARCHAR qui est du même type de données que l'argument string en entrée.

Exemples

Pour renvoyer la chaîne `''CAT''` avec des guillemets SIMPLES, utilisez l'exemple suivant.

```
SELECT QUOTE_LITERAL('''CAT''');

+---------------+
| quote_literal |
+---------------+
| '''CAT'''     |
+---------------+
```

Les exemples suivants utilisent les données de la table CATEGORY de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour renvoyer la colonne CATNAME entourée de guillemets simples, utilisez l'exemple suivant.

```
SELECT catid, QUOTE_LITERAL(catname)
FROM category
ORDER BY 1,2;

+-------+---------------+
| catid | quote_literal |
+-------+---------------+
|     1 | 'MLB'         |
|     2 | 'NHL'         |
|     3 | 'NFL'         |
|     4 | 'NBA'         |
|     5 | 'MLS'         |
|     6 | 'Musicals'    |
|     7 | 'Plays'       |
|     8 | 'Opera'       |
|     9 | 'Pop'         |
|    10 | 'Jazz'        |
|    11 | 'Classical'   |
+-------+---------------+
```

Pour renvoyer la colonne CATID entourée de guillemets simples, utilisez l'exemple suivant.

```
SELECT QUOTE_LITERAL(catid), catname
FROM category
ORDER BY 1,2;

+---------------+-----------+
| quote_literal |  catname  |
+---------------+-----------+
```

```
| '1'           | MLB       |
| '10'          | Jazz      |
| '11'          | Classical |
| '2'           | NHL       |
| '3'           | NFL       |
| '4'           | NBA       |
| '5'           | MLS       |
| '6'           | Musicals  |
| '7'           | Plays     |
| '8'           | Opera     |
| '9'           | Pop       |
+---------------+-----------+
```

Fonction REGEXP_COUNT

Recherche un modèle d'expression régulière dans une chaîne et renvoie un entier indiquant le nombre de fois où le modèle spécifié apparaît dans la chaîne. Si aucune correspondance n'est trouvée, la fonction renvoie `0`. Pour plus d'informations sur les expressions régulières, voir Opérateurs POSIX et Expression régulière dans Wikipedia.

Syntaxe

```
REGEXP_COUNT( source_string, pattern [, position [, parameters ] ] )
```

Arguments

source_string

Chaîne `CHAR` ou `VARCHAR`.

pattern

Chaîne littérale UTF-8 qui représente un modèle d'expression régulière. Pour de plus amples informations, veuillez consulter Opérateurs POSIX.

position

(Facultatif) `INTEGER` positif qui indique la position dans source_string où commencer la recherche. La position est basée sur le nombre de caractères, pas d'octets, de sorte que les caractères à plusieurs octets soient comptés comme des caractères seuls. L'argument par défaut est 1. Si position est inférieur à 1, la recherche commence au premier caractère de source_string. Si position est supérieur au nombre de caractères de source_string, le résultat est `0`.

parameters

(Facultatif) Un ou plusieurs littéraux de chaîne qui indiquent comment la fonction correspond au modèle. Les valeurs possibles sont les suivantes :

- c : réaliser une correspondance avec respect de la casse. Par défaut, la correspondance avec respect de la casse est utilisée.
- i : réaliser une correspondance avec non-respect de la casse.
- p – Interpréter le modèle avec le type d'expression PCRE (Perl Compatible Regular Expression). Pour plus d'informations sur le PCRE, consultez Expressions régulières compatibles avec Perl sur Wikipedia.

Type de retour

INTEGER

Exemples

Pour compter le nombre de fois qu'une séquence de trois lettres apparaît, utilisez l'exemple suivant.

```
SELECT REGEXP_COUNT('abcdefghijklmnopqrstuvwxyz', '[a-z]{3}');

+--------------+
| regexp_count |
+--------------+
|            8 |
+--------------+
```

Pour compter les occurrences de la chaîne FOX en utilisant une correspondance respectant la casse, utilisez l'exemple suivant.

```
SELECT REGEXP_COUNT('the fox', 'FOX', 1, 'i');

+--------------+
| regexp_count |
+--------------+
|            1 |
+--------------+
```

Pour utiliser un modèle écrit en PCRE pour localiser des mots contenant au moins un chiffre et une lettre minuscule, utilisez l'exemple suivant. Cet exemple utilise l'opérateur ?=, qui a une connotation

« anticipée » spécifique dans une PCRE. Cet exemple compte le nombre d'occurrences de ces mots, avec une correspondance avec respect de la casse.

```
SELECT REGEXP_COUNT('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 'p');

+--------------+
| regexp_count |
+--------------+
|            2 |
+--------------+
```

Pour utiliser un modèle écrit en PCRE pour localiser des mots contenant au moins un chiffre et une lettre minuscule, utilisez l'exemple suivant. Il utilise l'opérateur ?=, qui a une connotation spécifique au type PCRE. Cet exemple compte le nombre d'occurrences de ces mots, mais diffère de l'exemple précédent car il utilise une correspondance avec non-respect de la casse.

```
SELECT REGEXP_COUNT('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 'ip');

+--------------+
| regexp_count |
+--------------+
|            3 |
+--------------+
```

L'exemple suivant utilise les données de la table USERS de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour compter le nombre de fois que le nom de domaine de niveau supérieur est `org` ou `edu`, utilisez l'exemple suivant.

```
SELECT email, REGEXP_COUNT(email,'@[^.]*\.(org|edu)') FROM users
ORDER BY userid LIMIT 4;

+----------------------------------------------+--------------+
|                     email                    | regexp_count |
+----------------------------------------------+--------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu |            1 |
| Suspendisse.tristique@nonnisiAenean.edu       |            1 |
| amet.faucibus.ut@condimentumegetvolutpat.ca   |            0 |
```

```
| sed@lacusUtnec.ca                             |           0 |
+-----------------------------------------------+-------------+
```

Fonction REGEXP_INSTR

Recherche un modèle d'expression régulière dans une chaîne et renvoie un nombre entier qui indique la position de début de la sous-chaîne correspondante. Si aucune correspondance n'est trouvée, la fonction renvoie `0`. REGEXP_SUBSTR est similaire à la fonction POSITION, mais vous permet de rechercher un modèle d'expression régulière dans une chaîne. Pour plus d'informations sur les expressions régulières, voir Opérateurs POSIX et Expression régulière dans Wikipedia.

Syntaxe

```
REGEXP_INSTR( source_string, pattern [, position [, occurrence] [, option [, parameters
 ] ] ] ] )
```

Arguments

source_string

Expression de chaîne, comme un nom de colonne, à rechercher.

pattern

Chaîne littérale UTF-8 qui représente un modèle d'expression régulière. Pour de plus amples informations, veuillez consulter Opérateurs POSIX.

position

(Facultatif) `INTEGER` positif qui indique la position dans source_string où commencer la recherche. La position est basée sur le nombre de caractères, pas d'octets, de sorte que les caractères à plusieurs octets soient comptés comme des caractères seuls. L'argument par défaut est 1. Si position est inférieur à `1`, la recherche commence au premier caractère de source_string. Si position est supérieur au nombre de caractères de source_string, le résultat est `0`.

occurrence

(Facultatif) `INTEGER` positif qui indique quelle occurrence du modèle utiliser. REGEXP_INSTR ignore les *`occurrence`*`-1` premières correspondances. L'argument par défaut est 1. Si occurrence est inférieur à 1 ou supérieur au nombre de caractères dans source_string, la recherche est ignorée et le résultat est `0`.

option

(Facultatif) Valeur qui indique s'il faut renvoyer la position du premier caractère de la correspondance (`0`) ou celle du premier caractère après la fin de la correspondance (`1`). Une valeur non nulle est identique à `1`. La valeur par défaut est `0`.

parameters

(Facultatif) Un ou plusieurs littéraux de chaîne qui indiquent comment la fonction correspond au modèle. Les valeurs possibles sont les suivantes :

- c : réaliser une correspondance avec respect de la casse. Par défaut, la correspondance avec respect de la casse est utilisée.
- i : réaliser une correspondance avec non-respect de la casse.
- e : extraire une sous-chaîne à l'aide d'une sous-expression.

  Si pattern inclut une sous-expression, REGEXP_INSTR met en correspondance une sous-chaîne à l'aide de la première sous-expression incluse dans pattern. REGEXP_INSTR considère uniquement la première sous-expression ; les autres sous-expressions sont ignorées. Si le modèle n'inclut pas de sous-expression, REGEXP_INSTR ignore le paramètre « e ».
- p – Interpréter le modèle avec le type d'expression PCRE (Perl Compatible Regular Expression). Pour plus d'informations sur le PCRE, consultez Expressions régulières compatibles avec Perl sur Wikipedia.

Type de retour

Entier

Exemples

Les exemples suivants utilisent les données de la table USERS de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour rechercher le caractère `@` qui commence un nom de domaine et renvoyer la position de début de la première correspondance, utilisez l'exemple suivant.

```
SELECT email, REGEXP_INSTR(email, '@[^.]*')
FROM users
ORDER BY userid LIMIT 4;
```

```
+-----------------------------------------------+--------------+
|                     email                     | regexp_instr |
+-----------------------------------------------+--------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu |           21 |
| Suspendisse.tristique@nonnisiAenean.edu       |           22 |
| amet.faucibus.ut@condimentumegetvolutpat.ca   |           17 |
| sed@lacusUtnec.ca                             |            4 |
+-----------------------------------------------+--------------+
```

Pour rechercher des variantes du mot `Center` et renvoyer la position de début de la première correspondance, utilisez l'exemple suivant.

```
SELECT venuename, REGEXP_INSTR(venuename,'[cC]ent(er|re)$')
FROM venue
WHERE REGEXP_INSTR(venuename,'[cC]ent(er|re)$') > 0
ORDER BY venueid LIMIT 4;

+----------------------+--------------+
|       venuename      | regexp_instr |
+----------------------+--------------+
| The Home Depot Center |           16 |
| Izod Center           |            6 |
| Wachovia Center       |           10 |
| Air Canada Centre     |           12 |
+----------------------+--------------+
```

Pour rechercher la position de début de la première occurrence de la chaîne `FOX`, à l'aide d'une logique de correspondance respectant la casse, utilisez l'exemple suivant.

```
SELECT REGEXP_INSTR('the fox', 'FOX', 1, 1, 0, 'i');

+--------------+
| regexp_instr |
+--------------+
|            5 |
+--------------+
```

Pour utiliser un modèle écrit en PCRE pour localiser des mots contenant au moins un chiffre et une lettre minuscule, utilisez l'exemple suivant. Il utilise l'opérateur `?=`, qui a une connotation « anticipée » spécifique au type PCRE. Cet exemple montre comment trouver la position de départ du deuxième mot de ce type.

```
SELECT REGEXP_INSTR('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 0, 'p');

+--------------+
| regexp_instr |
+--------------+
|           21 |
+--------------+
```

Pour utiliser un modèle écrit en PCRE pour localiser des mots contenant au moins un chiffre et une lettre minuscule, utilisez l'exemple suivant. Il utilise l'opérateur ?=, qui a une connotation « anticipée » spécifique au type PCRE. Cet exemple recherche la position de départ du deuxième mot de ce type, mais diffère de l'exemple précédent car il utilise une correspondance avec non-respect de la casse.

```
SELECT REGEXP_INSTR('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 0, 'ip');

+--------------+
| regexp_instr |
+--------------+
|           15 |
+--------------+
```

Fonction REGEXP_REPLACE

Recherche un modèle d'expression régulière dans une chaîne et remplace chaque occurrence du modèle par la chaîne spécifiée. REGEXP_REPLACE est similaire à la Fonction REPLACE, mais vous permet de rechercher un modèle d'expression régulière dans une chaîne. Pour plus d'informations sur les expressions régulières, voir Opérateurs POSIX et Expression régulière dans Wikipedia.

REGEXP_REPLACE est similaire à la Fonction TRANSLATE et la Fonction REPLACE, sauf que TRANSLATE fait plusieurs remplacements de caractère unique et REPLACE remplace une chaîne entière par une autre chaîne, tandis que REGEXP_REPLACE vous permet de rechercher un modèle d'expression régulière dans une chaîne.

Syntaxe

```
REGEXP_REPLACE( source_string, pattern [, replace_string [ , position [, parameters
 ] ] ] )
```

Arguments

source_string

Expression de chaîne CHAR ou VARCHAR, comme un nom de colonne, à rechercher.

pattern

Chaîne littérale UTF-8 qui représente un modèle d'expression régulière. Pour de plus amples informations, veuillez consulter Opérateurs POSIX.

replace_string

(Facultatif) Expression de chaîne CHAR ou VARCHAR, comme un nom de colonne, qui va remplacer chaque occurrence de modèle. La valeur par défaut est une chaîne vide ("").

position

(Facultatif) Entier positif qui indique la position dans source_string où commencer la recherche. La position est basée sur le nombre de caractères, pas d'octets, de sorte que les caractères à plusieurs octets soient comptés comme des caractères seuls. L'argument par défaut est 1. Si position est inférieur à 1, la recherche commence au premier caractère de source_string. Si position est supérieure au nombre de caractères de source_string, le résultat est source_string.

parameters

(Facultatif) Un ou plusieurs littéraux de chaîne qui indiquent comment la fonction correspond au modèle. Les valeurs possibles sont les suivantes :

- c : réaliser une correspondance avec respect de la casse. Par défaut, la correspondance avec respect de la casse est utilisée.
- i : réaliser une correspondance avec non-respect de la casse.
- p – Interpréter le modèle avec le type d'expression PCRE (Perl Compatible Regular Expression). Pour plus d'informations sur le PCRE, consultez Expressions régulières compatibles avec Perl sur Wikipedia.

Type de retour

VARCHAR

Si pattern ou replace_string a la valeur NULL, la fonction renvoie NULL.

Exemples

Pour remplacer toutes les occurrences de la chaîne `FOX` dans la valeur `quick brown fox`, à l'aide d'une correspondance respectant la casse, utilisez l'exemple suivant.

```
SELECT REGEXP_REPLACE('the fox', 'FOX', 'quick brown fox', 1, 'i');

+---------------------+
|   regexp_replace    |
+---------------------+
| the quick brown fox |
+---------------------+
```

L'exemple suivant utilise un modèle écrit dans le type PCRE pour localiser des mots contenant au moins un chiffre et une lettre minuscule. Il utilise l'opérateur ?=, qui a une connotation « anticipée » spécifique au type PCRE. Pour remplacer chaque occurrence d'un tel mot par la valeur `[hidden]`, utilisez l'exemple suivant.

```
SELECT REGEXP_REPLACE('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 '[hidden]', 1, 'p');

+-------------------------------+
|        regexp_replace         |
+-------------------------------+
| [hidden] plain A1234 [hidden] |
+-------------------------------+
```

L'exemple suivant utilise un modèle écrit dans le type PCRE pour localiser des mots contenant au moins un chiffre et une lettre minuscule. Il utilise l'opérateur ?=, qui a une connotation « anticipée » spécifique au type PCRE. Pour remplacer chaque occurrence d'un tel mot par la valeur `[hidden]` (diffère de l'exemple précédent car une correspondance ne respectant pas la casse est utilisée), utilisez l'exemple suivant.

```
SELECT REGEXP_REPLACE('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 '[hidden]', 1, 'ip');

+---------------------------------+
|          regexp_replace         |
+---------------------------------+
| [hidden] plain [hidden] [hidden] |
```

```
+---------------------------------+
```

Les exemples suivants utilisent les données de la table USERS de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour supprimer le caractère @ et le nom de domaine des adresses e-mail, utilisez l'exemple suivant.

```
SELECT email, REGEXP_REPLACE(email, '@.*\\.(org|gov|com|edu|ca)$')
FROM users
ORDER BY userid LIMIT 4;

+-----------------------------------------------+----------------------+
|                     email                     |    regexp_replace    |
+-----------------------------------------------+----------------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu | Etiam.laoreet.libero |
| Suspendisse.tristique@nonnisiAenean.edu       | Suspendisse.tristique |
| amet.faucibus.ut@condimentumegetvolutpat.ca   | amet.faucibus.ut     |
| sed@lacusUtnec.ca                             | sed                  |
+-----------------------------------------------+----------------------+
```

Pour remplacer les noms de domaine des adresses e-mail par `internal.company.com`, utilisez l'exemple suivant.

```
SELECT email, REGEXP_REPLACE(email, '@.*\\.[[:alpha:]]{2,3}','@internal.company.com')
FROM users
ORDER BY userid LIMIT 4;

+-----------------------------------------------
+-------------------------------------------+
|                     email                     |               regexp_replace
      |
+-----------------------------------------------
+-------------------------------------------+
| Etiam.laoreet.libero@sodalesMaurisblandit.edu |
 Etiam.laoreet.libero@internal.company.com  |
| Suspendisse.tristique@nonnisiAenean.edu       |
 Suspendisse.tristique@internal.company.com |
| amet.faucibus.ut@condimentumegetvolutpat.ca   | amet.faucibus.ut@internal.company.com
      |
| sed@lacusUtnec.ca                             | sed@internal.company.com
      |
```

```
+-------------------------------------------
+-----------------------------------------+
```

Fonction REGEXP_SUBSTR

Renvoie les caractères d'une chaîne en y recherchant un modèle d'expression régulière. REGEXP_SUBSTR est similaire à la fonction Fonction SUBSTRING, mais vous permet de rechercher un modèle d'expression régulière dans une chaîne. Si la fonction ne trouve pas correspondance entre l'expression régulière et aucun caractère de la chaîne, elle renvoie une chaîne vide. Pour plus d'informations sur les expressions régulières, voir Opérateurs POSIX et Expression régulière dans Wikipedia.

Syntaxe

```
REGEXP_SUBSTR( source_string, pattern [, position [, occurrence [, parameters ] ] ] )
```

Arguments

source_string

Expression de chaîne à rechercher.

pattern

Chaîne littérale UTF-8 qui représente un modèle d'expression régulière. Pour de plus amples informations, veuillez consulter Opérateurs POSIX.

position

Nombre entier positif qui indique à quel endroit de source_string commencer la recherche. La position est basée sur le nombre de caractères, pas d'octets, de sorte que les caractères à plusieurs octets soient comptés comme des caractères seuls. La valeur par défaut est 1. Si position est inférieur à 1, la recherche commence au premier caractère de source_string. Si position est supérieure au nombre de caractères de source_string, le résultat est une chaîne vide ("").

occurrence

Nombre entier positif qui indique quelle occurrence du modèle utiliser. REGEXP_SUBSTR ignore les occurrence -1 premières correspondances. La valeur par défaut est 1. Si l'occurrence est inférieure à 1 ou supérieure au nombre de caractères de source_string, la recherche est ignorée et le résultat est vide.

parameters

Un ou plusieurs littéraux de chaîne qui indiquent comment la fonction correspond au modèle. Les valeurs possibles sont les suivantes :

- c : réaliser une correspondance avec respect de la casse. Par défaut, la correspondance avec respect de la casse est utilisée.
- i : réaliser une correspondance avec non-respect de la casse.
- e : extraire une sous-chaîne à l'aide d'une sous-expression.

  Si pattern inclut une sous-expression, REGEXP_SUBSTR met en correspondance une sous-chaîne à l'aide de la première sous-expression incluse dans pattern. Une sous-expression est une expression dans le modèle qui est mise entre parenthèses. Par exemple, le modèle `'This is a (\\w+)'` met en correspondance la première expression avec la chaîne `'This is a '` suivie d'un mot. Au lieu de renvoyer le modèle, REGEXP_SUBSTR avec le paramètre e renvoie uniquement la chaîne contenue dans la sous-expression.

  REGEXP_SUBSTR considère uniquement la première sous-expression ; les autres sous-expressions sont ignorées. Si le modèle n'inclut pas de sous-expression, REGEXP_SUBSTR ignore le paramètre « e ».
- p – Interpréter le modèle avec le type d'expression PCRE (Perl Compatible Regular Expression). Pour plus d'informations sur le PCRE, consultez Expressions régulières compatibles avec Perl sur Wikipedia.

Type de retour

VARCHAR

Exemples

L'exemple suivant renvoie la partie d'une adresse e-mail comprise entre le caractère @ et l'extension du domaine. Les données `users` interrogées proviennent des exemples de données Amazon Redshift. Pour de plus amples informations, veuillez consulter Exemple de base de données.

```
SELECT email, regexp_substr(email,'@[^.]*')
FROM users
ORDER BY userid LIMIT 4;

                     email                     |      regexp_substr
-----------------------------------------------+--------------------------
```

```
 Suspendisse.tristique@nonnisiAenean.edu       | @nonnisiAenean
 amet.faucibus.ut@condimentumegetvolutpat.ca   | @condimentumegetvolutpat
 sed@lacusUtnec.ca                             | @lacusUtnec
 Cum@accumsan.com                              | @accumsan
```

L'exemple suivant renvoie la partie de l'entrée correspondant à la première occurrence de la chaîne `FOX` à l'aide d'une correspondance ne respectant pas la casse.

```
SELECT regexp_substr('the fox', 'FOX', 1, 1, 'i');

 regexp_substr
---------------
 fox
```

L'exemple suivant renvoie la partie de l'entrée correspondant à la deuxième occurrence de la chaîne `FOX` à l'aide d'une correspondance ne respectant pas la casse. Le résultat est vide (non nul, longueur 0) car il n'y a pas de deuxième occurrence.

```
SELECT regexp_substr('the fox', 'FOX', 1, 2, 'i');

 regexp_substr
---------------

```

L'exemple suivant renvoie la première partie de l'entrée qui commence par des lettres minuscules. Il est fonctionnellement identique à la même instruction SELECT sans le paramètre `c`.

```
SELECT regexp_substr('THE SECRET CODE IS THE LOWERCASE PART OF 1931abc0EZ.', '[a-z]+',
 1, 1, 'c');

 regexp_substr
---------------
 abc
```

L'exemple suivant utilise un modèle écrit dans le type PCRE pour localiser des mots contenant au moins un chiffre et une lettre minuscule. Il utilise l'opérateur ?=, qui a une connotation « anticipée » spécifique au type PCRE. Cet exemple renvoie la partie de l'entrée correspondant au deuxième mot de ce type.

```
SELECT regexp_substr('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 'p');
```

```
 regexp_substr
---------------
 a1234
```

L'exemple suivant utilise un modèle écrit dans le type PCRE pour localiser des mots contenant au moins un chiffre et une lettre minuscule. Il utilise l'opérateur ?=, qui a une connotation « anticipée » spécifique au type PCRE. Cet exemple renvoie la partie de l'entrée correspondant au deuxième mot de ce type, mais diffère de l'exemple précédent car il utilise une correspondance avec non-respect de la casse.

```
SELECT regexp_substr('passwd7 plain A1234 a1234', '(?=[^ ]*[a-z])(?=[^ ]*[0-9])[^ ]+',
 1, 2, 'ip');

 regexp_substr
---------------
 A1234
```

L'exemple suivant utilise une sous-expression pour rechercher la deuxième chaîne correspondant au modèle `'this is a (\\w+)'` à l'aide d'une correspondance avec respect de la casse. Il renvoie la sous-expression entre parenthèses.

```
SELECT regexp_substr(
               'This is a cat, this is a dog. This is a mouse.',
               'this is a (\\w+)', 1, 2, 'ie');

 regexp_substr
---------------
 dog
```

Fonction REPEAT

Répète une chaîne le nombre de fois spécifié. Si le paramètre d'entrée est numérique, REPEAT le traite sous forme de chaîne.

Synonyme de Fonction REPLICATE.

Syntaxe

```
REPEAT(string, integer)
```

Arguments

string

Le premier paramètre d'entrée est la chaîne à répéter.

integer

Le deuxième paramètre est un `INTEGER` indiquant combien de fois répéter la chaîne.

Type de retour

VARCHAR

Exemples

L'exemple suivant utilise les données de la table CATEGORY de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour répéter la valeur de la colonne CATID dans la table CATEGORY à trois reprises, utilisez l'exemple suivant.

```
SELECT catid, REPEAT(catid,3)
FROM category
ORDER BY 1,2;

+-------+--------+
| catid | repeat |
+-------+--------+
|     1 |    111 |
|     2 |    222 |
|     3 |    333 |
|     4 |    444 |
|     5 |    555 |
|     6 |    666 |
|     7 |    777 |
|     8 |    888 |
|     9 |    999 |
|    10 | 101010 |
|    11 | 111111 |
+-------+--------+
```

Fonction REPLACE

Remplace toutes les occurrences d'un jeu de caractères au sein d'une chaîne existante par d'autres caractères spécifiés.

REPLACE est similaire à la Fonction TRANSLATE et la Fonction REGEXP_REPLACE, sauf que TRANSLATE fait plusieurs remplacements de caractère unique et REGEXP_REPLACE vous permet de rechercher un modèle d'expression régulière dans une chaîne, tandis que REPLACE remplace une chaîne entière par une autre chaîne.

Syntaxe

```
REPLACE(string, old_chars, new_chars)
```

Arguments

string

Chaîne CHAR ou VARCHAR à rechercher

old_chars

Chaîne CHAR ou VARCHAR à remplacer.

new_chars

Nouvelle chaîne CHAR ou VARCHAR remplaçant l'ancienne chaîne old_string.

Type de retour

VARCHAR

Si old_chars ou new_chars a la valeur NULL, le retour est NULL.

Exemples

L'exemple suivant utilise les données de la table CATEGORY de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour convertir la chaîne Shows en Theatre dans le champ CATGROUP, utilisez l'exemple suivant.

```
SELECT catid, catgroup, REPLACE(catgroup, 'Shows', 'Theatre')
FROM category
```

```
ORDER BY 1,2,3;

+-------+----------+----------+
| catid | catgroup | replace  |
+-------+----------+----------+
|     1 | Sports   | Sports   |
|     2 | Sports   | Sports   |
|     3 | Sports   | Sports   |
|     4 | Sports   | Sports   |
|     5 | Sports   | Sports   |
|     6 | Shows    | Theatre  |
|     7 | Shows    | Theatre  |
|     8 | Shows    | Theatre  |
|     9 | Concerts | Concerts |
|    10 | Concerts | Concerts |
|    11 | Concerts | Concerts |
+-------+----------+----------+
```

Fonction REPLICATE

Synonyme de la fonction REPEAT.

Consultez Fonction REPEAT.

Fonction REVERSE

La fonction REVERSE s'applique à une chaîne et renvoie les caractères dans l'ordre inverse. Par exemple, `reverse('abcde')` renvoie `edcba`. Cette fonction s'applique aux types de données numérique et de date, ainsi qu'aux types de données de caractère. Toutefois, dans la plupart des cas, elle a une valeur pratique pour les chaînes de caractères.

Syntaxe

```
REVERSE( expression )
```

Argument

expression

Expression avec un type de données de caractère, date, horodatage ou numérique qui représente la cible de l'inversion de caractères. Toutes les expressions sont implicitement converties en chaînes `VARCHAR`. Les espaces de fin des chaînes `CHAR` sont ignorés.

Type de retour

VARCHAR

Exemples

Les exemples suivants utilisent les données des tables USERS et SALES de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour sélectionner cinq noms de ville distincts et leur noms inversés correspondants à partir de la table USERS, utilisez l'exemple suivant.

```
SELECT DISTINCT city AS cityname, REVERSE(cityname)
FROM users
ORDER BY city LIMIT 5;

+----------+----------+
| cityname | reverse  |
+----------+----------+
| Aberdeen | needrebA |
| Abilene  | enelibA  |
| Ada      | adA      |
| Agat     | tagA     |
| Agawam   | mawagA   |
+----------+----------+
```

Pour sélectionner cinq ventes IDs et leur transformation inversée correspondante IDs sous forme de chaînes de caractères, utilisez l'exemple suivant.

```
SELECT salesid, REVERSE(salesid)
FROM sales
ORDER BY salesid DESC LIMIT 5;

+---------+---------+
| salesid | reverse |
+---------+---------+
|  172456 |  654271 |
|  172455 |  554271 |
|  172454 |  454271 |
|  172453 |  354271 |
|  172452 |  254271 |
+---------+---------+
```

Fonction RTRIM

La fonction RTRIM supprime un ensemble spécifié de caractères à partir de la fin d'une chaîne. Supprime la chaîne la plus longue ne contenant que des caractères de la liste des caractères supprimés. La suppression est terminée lorsqu'un caractère supprimé n'apparaît pas dans la chaîne d'entrée.

Syntaxe

```
RTRIM( string, trim_chars )
```

Arguments

string

Une colonne de chaîne, une expression ou un littéral de chaîne à supprimer.

trim_chars

Colonne de chaîne, expression ou littéral de chaîne représentant les caractères à supprimer à la fin de la chaîne. Si la valeur n'est pas spécifiée, un espace est utilisé comme caractère de séparation.

Type de retour

Chaîne qui a le même type de données que l'argument string.

exemple

L'exemple suivant tronque les espaces de début et de fin de la chaîne `' abc '` :

```
select '     abc    ' as untrim, rtrim('     abc    ') as trim;

untrim    | trim
----------+------
   abc    |    abc
```

L'exemple suivant supprime les chaînes `'xyz'` de fin de la chaîne `'xyzaxyzbxyzcxyz'`. Les occurrences de fin de `'xyz'` sont supprimées, mais celles qui se trouvent à l'intérieur de la chaîne sont conservées.

```
select 'xyzaxyzbxyzcxyz' as untrim,
rtrim('xyzaxyzbxyzcxyz', 'xyz') as trim;

     untrim      |   trim
-----------------+-----------
 xyzaxyzbxyzcxyz | xyzaxyzbxyzc
```

L'exemple suivant supprime les parties de fin de la chaîne `'setuphistorycassettes'` qui correspondent à l'un des caractères de la liste trim_chars `'tes'`. Tout caractère `t`, `e` ou `s` précédant un autre caractère qui ne figure pas dans la liste trim_chars à la fin de la chaîne d'entrée est supprimé.

```
SELECT rtrim('setuphistorycassettes', 'tes');

     rtrim
-----------------
 setuphistoryca
```

L'exemple suivant tronque les caractères « Park » à la fin de VENUENAME le cas échéant :

```
select venueid, venuename, rtrim(venuename, 'Park')
from venue
order by 1, 2, 3
limit 10;

venueid |         venuename          |          rtrim
--------+----------------------------+-------------------------
      1 | Toyota Park                | Toyota
      2 | Columbus Crew Stadium      | Columbus Crew Stadium
      3 | RFK Stadium                | RFK Stadium
      4 | CommunityAmerica Ballpark  | CommunityAmerica Ballp
      5 | Gillette Stadium           | Gillette Stadium
      6 | New York Giants Stadium    | New York Giants Stadium
      7 | BMO Field                  | BMO Field
      8 | The Home Depot Center      | The Home Depot Cente
      9 | Dick's Sporting Goods Park | Dick's Sporting Goods
     10 | Pizza Hut Park             | Pizza Hut
```

Notez que RTRIM supprime les caractères `P`, `a`, `r` ou `k` lorsqu'ils apparaissent à la fin d'un VENUENAME.

Fonction SOUNDEX

La fonction SOUNDEX renvoie la valeur Soundex américaine consistant en la première lettre de la chaîne en entrée, suivie d'un codage à 3 chiffres des sons qui représentent la prononciation anglaise de la chaîne que vous spécifiez. Par exemple, `Smith` et `Smyth` ont la même valeur Soundex.

Syntaxe

```
SOUNDEX(string)
```

Arguments

string

Vous spécifiez une chaîne `CHAR` ou `VARCHAR` que vous souhaitez convertir en une valeur de code Soundex américain.

Type de retour

VARCHAR(4)

Notes d'utilisation

La fonction SOUNDEX convertit uniquement les caractères alphabétiques en minuscules et majuscules ASCII, y compris a–z et A–Z. SOUNDEX ignore les autres caractères. SOUNDEX renvoie une valeur Soundex unique pour une chaîne de mots multiples séparés par des espaces.

```
SELECT SOUNDEX('AWS Amazon');

+---------+
| soundex |
+---------+
| A252    |
+---------+
```

SOUNDEX renvoie une chaîne vide si la chaîne d'entrée ne contient pas de lettres anglaises.

```
SELECT SOUNDEX('+-*/%');

+---------+
| soundex |
```

```
+---------+
|         |
+---------+
```

Exemples

Pour renvoyer la valeur Soundex pour `Amazon`, utilisez l'exemple suivant.

```
SELECT SOUNDEX('Amazon');

+---------+
| soundex |
+---------+
| A525    |
+---------+
```

Pour renvoyer la valeur Soundex pour `smith` et `smyth`, utilisez l'exemple suivant. Notez que les valeurs Soundex sont identiques.

```
SELECT SOUNDEX('smith'), SOUNDEX('smyth');

+-------+-------+
| smith | smyth |
+-------+-------+
| S530  | S530  |
+-------+-------+
```

Fonction SPLIT_PART

Divise une chaîne sur le délimiteur spécifié et renvoie la partie à la position spécifiée.

Syntaxe

```
SPLIT_PART(string, delimiter, position)
```

Arguments

string

Colonne de chaîne, expression ou littéral de chaîne à fractionner. La chaîne peut être CHAR ou VARCHAR.

delimiter

Chaîne de délimiteur indiquant les sections de la chaîne d'entrée.

Si delimiter est un littéral, mettez-le entre guillemets simples.

position

Position de la partie de chaîne à renvoyer (à partir de 1). Doit être un nombre entier supérieur à 0. Si la valeur de position est supérieure au nombre de parties de chaîne, SPLIT_PART renvoie une chaîne vide. Si délimiteur est introuvable dans chaîne, alors la valeur renvoyée contient le contenu de la partie spécifiée, qui pourrait être la chaîne entière ou une valeur vide.

Type de retour

Chaîne CHAR ou VARCHAR, la même que le paramètre string.

Exemples

L'exemple suivant fractionne un littéral de chaîne en différentes parties en utilisant le délimiteur $ et renvoie la seconde partie.

```
select split_part('abc$def$ghi','$',2)

split_part
----------
def
```

L'exemple suivant fractionne un littéral de chaîne en différentes parties en utilisant le délimiteur $. Il renvoie une chaîne vide, car la partie 4 est introuvable.

```
select split_part('abc$def$ghi','$',4)

split_part
----------

```

L'exemple suivant fractionne un littéral de chaîne en différentes parties en utilisant le délimiteur #. Il renvoie la chaîne entière, qui correspond à la première partie, car le délimiteur est introuvable.

```
select split_part('abc$def$ghi','#',1)
```

```
split_part
------------
abc$def$ghi
```

L'exemple suivant divise le champ d'horodatage LISTTIME en composants d'année, de mois et de date.

```
select listtime, split_part(listtime,'-',1) as year,
split_part(listtime,'-',2) as month,
split_part(split_part(listtime,'-',3),' ',1) as day
from listing limit 5;

      listtime       | year | month | day
---------------------+------+-------+------
 2008-03-05 12:25:29 | 2008 | 03    | 05
 2008-09-09 08:03:36 | 2008 | 09    | 09
 2008-09-26 05:43:12 | 2008 | 09    | 26
 2008-10-04 02:00:30 | 2008 | 10    | 04
 2008-01-06 08:33:11 | 2008 | 01    | 06
```

L'exemple suivant sélectionne le champ d'horodatage LISTTIME et le divise sur le caractère `'-'` pour obtenir le mois (la deuxième partie de la chaîne LISTTIME), puis compte le nombre d'entrées de chaque mois :

```
select split_part(listtime,'-',2) as month, count(*)
from listing
group by split_part(listtime,'-',2)
order by 1, 2;

 month | count
-------+-------
    01 | 18543
    02 | 16620
    03 | 17594
    04 | 16822
    05 | 17618
    06 | 17158
    07 | 17626
    08 | 17881
    09 | 17378
    10 | 17756
```

```
    11 | 12912
    12 | 4589
```

Fonction STRPOS

Renvoie la position d'une sous-chaîne dans une chaîne spécifiée.

Consultez Fonction CHARINDEX et Fonction POSITION pour des fonctions similaires.

Syntaxe

```
STRPOS(string, substring )
```

Arguments

string

Le premier paramètre d'entrée est la chaîne CHAR ou VARCHAR à rechercher.

substring

Le deuxième paramètre est la sous-chaîne à rechercher dans la chaîne.

Type de retour

INTEGER

La fonction STRPOS renvoie un INTEGER correspondant à la position de l'argument substring (de base 1, pas de base 0). La position est basée sur le nombre de caractères, pas d'octets, de sorte que les caractères à plusieurs octets soient comptés comme des caractères seuls.

Notes d'utilisation

STRPOS renvoie 0 si substring n'est pas trouvé dans string.

```
SELECT STRPOS('dogfish', 'fist');

+--------+
| strpos |
+--------+
|      0 |
```

```
+--------+
```

Exemples

Pour montrer la position de `fish` dans `dogfish`, utilisez l'exemple suivant.

```
SELECT STRPOS('dogfish', 'fish');

+--------+
| strpos |
+--------+
|      4 |
+--------+
```

L'exemple suivant utilise les données de la table SALES de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour renvoyer le nombre de transactions de vente avec une COMMISSION de plus de 999,00 à partir de la table SALES, utilisez l'exemple suivant.

```
SELECT DISTINCT STRPOS(commission, '.'),
COUNT (STRPOS(commission, '.'))
FROM sales
WHERE STRPOS(commission, '.') > 4
GROUP BY STRPOS(commission, '.')
ORDER BY 1, 2;

+--------+-------+
| strpos | count |
+--------+-------+
|      5 |   629 |
+--------+-------+
```

Fonction STRTOL

Convertit une expression de chaîne d'un nombre de la base spécifiée en la valeur de nombre entier équivalente. La valeur convertie doit est comprise dans la plage de 64 bits signée.

Syntaxe

```
STRTOL(num_string, base)
```

Arguments

num_string

Expression de chaîne d'un nombre à convertir. Si num_string est vide ('') ou commence par le caractère null ('\0'), la valeur convertie est 0. Si num_string est une colonne contenant une valeur NULL, STRTOL renvoie NULL. La chaîne peut commencer par n'importe quelle quantité d'espace vide, éventuellement suivie d'un signe plus « + » ou moins « - » pour indiquer si elle est positive ou négative. La valeur par défaut est « + ». Si la base est 16, la chaîne peut éventuellement commencer par « 0x ».

base

INTEGER entre 2 et 36.

Type de retour

BIGINT

Si num_string a pour valeur null, la fonction renvoie NULL.

Exemples

Pour convertir des paires de valeur de chaîne et de base en entiers, utilisez les exemples suivants.

```
SELECT STRTOL('0xf',16);

+--------+
| strtol |
+--------+
|     15 |
+--------+

SELECT STRTOL('abcd1234',16);

+------------+
|   strtol   |
+------------+
| 2882343476 |
+------------+
```

```
SELECT STRTOL('1234567', 10);

+---------+
| strtol  |
+---------+
| 1234567 |
+---------+

SELECT STRTOL('1234567', 8);

+--------+
| strtol |
+--------+
| 342391 |
+--------+

SELECT STRTOL('110101', 2);

+--------+
| strtol |
+--------+
|     53 |
+--------+

SELECT STRTOL('\0', 2);

+--------+
| strtol |
+--------+
|      0 |
+--------+
```

Fonction SUBSTRING

Renvoie le sous-ensemble d'une chaîne sur la base de la position de départ spécifiée.

Si l'entrée est une chaîne de caractères, la position de départ et le nombre de caractères extraits sont basés sur les caractères, pas les octets, afin que les caractères à plusieurs octets soient comptés comme des caractères uniques. Si l'entrée est une expression binaire, la position de départ et la sous-chaîne extraite sont basées sur des octets. Vous ne pouvez pas spécifier de longueur négative, mais vous pouvez spécifier une position de début négative.

Syntaxe

```
SUBSTRING(character_string FROM start_position [ FOR number_characters ] )
```

```
SUBSTRING(character_string, start_position, number_characters )
```

```
SUBSTRING(binary_expression, start_byte, number_bytes )
```

```
SUBSTRING(binary_expression, start_byte )
```

Arguments

character_string

Chaîne à rechercher. Les types de données non-caractères sont traités comme une chaîne.

start_position

Position au sein de la chaîne à laquelle commencer l'extraction, à partir de 1. La position de début start_position est basée sur le nombre de caractères, pas d'octets, de sorte que les caractères à plusieurs octets soient comptés comme des caractères seuls. Ce numéro peut être négatif.

number_characters

Nombre de caractères à extraire (longueur de la sous-chaîne). Le nombre de caractères number_characters est basé sur le nombre de caractères, pas d'octets, de sorte que les caractères à plusieurs octets soient comptés comme des caractères seuls. Ce numéro ne peut pas être négatif.

binary_expression

binary_expression du type de données VARBYTE à rechercher.

start_byte

Position au sein de l'expression binaire à laquelle commencer l'extraction, à partir de 1. Ce numéro peut être négatif.

number_bytes

Nombre d'octets à extraire, c'est-à-dire la longueur de la sous-chaîne. Ce numéro ne peut pas être négatif.

Type de retour

VARCHAR ou VARBYTE selon l'entrée.

Notes d'utilisation

Vous trouverez ci-dessous quelques exemples de la façon dont vous pouvez utiliser start_position et number_characters pour extraire des sous-chaînes à partir de différentes positions d'une chaîne.

L'exemple suivant renvoie une chaîne de quatre caractères commençant par le sixième caractère.

```
select substring('caterpillar',6,4);
substring
-----------
pill
(1 row)
```

Si start_position + number_characters dépasse la longueur de la chaîne, SUBSTRING renvoie une sous-chaîne commençant par start_position jusqu'à la fin de la chaîne. Par exemple :

```
select substring('caterpillar',6,8);
substring
-----------
pillar
(1 row)
```

Si `start_position` est négatif ou égal à 0, la fonction SUBSTRING renvoie une sous-chaîne commençant au premier caractère de la chaîne d'une longueur de `start_position` + `number_characters` -1. Par exemple :

```
select substring('caterpillar',-2,6);
substring
-----------
cat
(1 row)
```

Si `start_position` + `number_characters` -1 est inférieur ou égal à zéro, SUBSTRING renvoie une chaîne vide. Par exemple :

```
select substring('caterpillar',-5,4);
```

```
substring
-----------

(1 row)
```

Exemples

L'exemple suivant renvoie le mois de la chaîne LISTTIME dans la table LISTING :

```
select listid, listtime,
substring(listtime, 6, 2) as month
from listing
order by 1, 2, 3
limit 10;

 listid |      listtime       | month
--------+---------------------+-------
      1 | 2008-01-24 06:43:29 | 01
      2 | 2008-03-05 12:25:29 | 03
      3 | 2008-11-01 07:35:33 | 11
      4 | 2008-05-24 01:18:37 | 05
      5 | 2008-05-17 02:29:11 | 05
      6 | 2008-08-15 02:08:13 | 08
      7 | 2008-11-15 09:38:15 | 11
      8 | 2008-11-09 05:07:30 | 11
      9 | 2008-09-09 08:03:36 | 09
     10 | 2008-06-17 09:44:54 | 06
(10 rows)
```

L'exemple suivant est le même que ci-dessus, mais utilise l'option FROM...FOR :

```
select listid, listtime,
substring(listtime from 6 for 2) as month
from listing
order by 1, 2, 3
limit 10;

 listid |      listtime       | month
--------+---------------------+-------
      1 | 2008-01-24 06:43:29 | 01
      2 | 2008-03-05 12:25:29 | 03
      3 | 2008-11-01 07:35:33 | 11
      4 | 2008-05-24 01:18:37 | 05
```

```
      5 | 2008-05-17 02:29:11 | 05
      6 | 2008-08-15 02:08:13 | 08
      7 | 2008-11-15 09:38:15 | 11
      8 | 2008-11-09 05:07:30 | 11
      9 | 2008-09-09 08:03:36 | 09
     10 | 2008-06-17 09:44:54 | 06
(10 rows)
```

Vous ne pouvez pas utiliser SUBSTRING pour extraire de manière prévisible le préfixe d'une chaîne pouvant contenir des caractères à plusieurs octets, car vous devez spécifier la longueur d'une chaîne de plusieurs octets basée sur le nombre d'octets, pas sur le nombre de caractères. Pour extraire le segment de début d'une chaîne en fonction de la longueur en octets, vous pouvez utiliser la fonction CAST sur la chaîne au format VARCHAR(byte_length) pour tronquer la chaîne, où byte_length est la longueur requise. L'exemple suivant extrait les 5 premiers octets de la chaîne `'Fourscore and seven'`.

```
select cast('Fourscore and seven' as varchar(5));

varchar
-------
Fours
```

L'exemple suivant illustre une position de départ négative d'une valeur binaire `abc`. Comme la position de départ est -3, la sous-chaîne est extraite du début de la valeur binaire. Le résultat est automatiquement affiché sous forme de représentation hexadécimale de la sous-chaîne binaire.

```
select substring('abc'::varbyte, -3);

 substring
-----------
 616263
```

L'exemple suivant montre un 1 pour la position de départ d'une valeur binaire `abc`. Comme aucune longueur n'est spécifiée, la chaîne est extraite de la position de départ jusqu'à la fin de la chaîne. Le résultat est automatiquement affiché sous forme de représentation hexadécimale de la sous-chaîne binaire.

```
select substring('abc'::varbyte, 1);

 substring
```

```
-----------
 616263
```

L'exemple suivant montre un 3 pour la position de départ d'une valeur binaire `abc`. Comme aucune longueur n'est spécifiée, la chaîne est extraite de la position de départ jusqu'à la fin de la chaîne. Le résultat est automatiquement affiché sous forme de représentation hexadécimale de la sous-chaîne binaire.

```
select substring('abc'::varbyte, 3);

 substring
-----------
 63
```

L'exemple suivant montre un 2 pour la position de départ d'une valeur binaire `abc`. La chaîne est extraite de la position de départ à la position 10, mais la fin de la chaîne est à la position 3. Le résultat est automatiquement affiché sous forme de représentation hexadécimale de la sous-chaîne binaire.

```
select substring('abc'::varbyte, 2, 10);

 substring
-----------
 6263
```

L'exemple suivant montre un 2 pour la position de départ d'une valeur binaire `abc`. La chaîne est extraite de la position de départ pour 1 octet. Le résultat est automatiquement affiché sous forme de représentation hexadécimale de la sous-chaîne binaire.

```
select substring('abc'::varbyte, 2, 1);

 substring
-----------
 62
```

L'exemple suivant renvoie le prénom `Ana` qui apparaît après le dernier espace de la chaîne d'entrée `Silva, Ana`.

```
select reverse(substring(reverse('Silva, Ana'), 1, position(' ' IN reverse('Silva,
 Ana'))))
```

```
 reverse
-----------
 Ana
```

Fonction TEXTLEN

Synonyme de la fonction LEN.

Consultez Fonction LEN.

Fonction TRANSLATE

Pour une expression données, remplace toutes les occurrences de caractères spécifiés par des produits de remplacement spécifiés. Les caractères existants sont mappés à des caractères de remplacement en fonction de leurs positions dans les arguments characters_to_replace et characters_to_substitute. Si le nombre de caractères spécifiés dans l'argument characters_to_replace est supérieur à celui de l'argument characters_to_substitute, les caractères supplémentaires depuis l'argument characters_to_replace sont omis dans la valeur de retour.

TRANSLATE est similaire à la Fonction REPLACE et la Fonction REGEXP_REPLACE, sauf que REPLACE remplace une chaîne entière par une autre chaîne et que REGEXP_REPLACE vous permet de rechercher un modèle d'expression régulière dans une chaîne, tandis que TRANSLATE fait plusieurs remplacements de caractère unique.

Si un argument a la valeur null, le retour est NULL.

Syntaxe

```
TRANSLATE( expression, characters_to_replace, characters_to_substitute )
```

Arguments

expression

Expression à traduire.

characters_to_replace

Chaîne contenant les caractères à remplacer.

characters_to_substitute

Chaîne contenant les caractères à remplacer.

Type de retour

VARCHAR

Exemples

Pour remplacer plusieurs caractères dans une chaîne, utilisez l'exemple suivant.

```
SELECT TRANSLATE('mint tea', 'inea', 'osin');

+-----------+
| translate |
+-----------+
| most tin  |
+-----------+
```

Les exemples suivants utilisent les données de la table USERS de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour remplacer le signe arobase (@) par un point dans toutes les valeurs d'une colonne, utilisez l'exemple suivant.

```
SELECT email, TRANSLATE(email, '@', '.') as obfuscated_email
FROM users LIMIT 10;

+--------------------------------------+--------------------------------------+
|                email                 |           obfuscated_email           |
+--------------------------------------+--------------------------------------+
| Cum@accumsan.com                     | Cum.accumsan.com                     |
| lorem.ipsum@Vestibulumante.com       | lorem.ipsum.Vestibulumante.com       |
| non.justo.Proin@ametconsectetuer.edu | non.justo.Proin.ametconsectetuer.edu |
| non.ante.bibendum@porttitortellus.org | non.ante.bibendum.porttitortellus.org |
| eros@blanditatnisi.org               | eros.blanditatnisi.org               |
| augue@Donec.ca                       | augue.Donec.ca                       |
| cursus@pedeacurna.edu                | cursus.pedeacurna.edu                |
| at@Duis.com                          | at.Duis.com                          |
| quam@facilisisvitaeorci.ca           | quam.facilisisvitaeorci.ca           |
| mi.lorem@nunc.edu                    | mi.lorem.nunc.edu                    |
```

```
+--------------------------------------+--------------------------------------+
```

Pour remplacer des espaces par des traits de soulignement et supprimer les points de toutes les valeurs d'une colonne, utilisez l'exemple suivant.

```
SELECT city, TRANSLATE(city, ' .', '_')
FROM users
WHERE city LIKE 'Sain%' OR city LIKE 'St%'
GROUP BY city
ORDER BY city;

+----------------+---------------+
|      city      |   translate   |
+----------------+---------------+
| Saint Albans   | Saint_Albans  |
| Saint Cloud    | Saint_Cloud   |
| Saint Joseph   | Saint_Joseph  |
| Saint Louis    | Saint_Louis   |
| Saint Paul     | Saint_Paul    |
| St. George     | St_George     |
| St. Marys      | St_Marys      |
| St. Petersburg | St_Petersburg |
| Stafford       | Stafford      |
| Stamford       | Stamford      |
| Stanton        | Stanton       |
| Starkville     | Starkville    |
| Statesboro     | Statesboro    |
| Staunton       | Staunton      |
| Steubenville   | Steubenville  |
| Stevens Point  | Stevens_Point |
| Stillwater     | Stillwater    |
| Stockton       | Stockton      |
| Sturgis        | Sturgis       |
+----------------+---------------+
```

Fonction TRIM

Tronque une chaîne en supprimant les espaces ou les caractères spécifiés.

Syntaxe

```
TRIM( [ BOTH | LEADING | TRAILING ] [trim_chars FROM ] string )
```

Arguments

BOTH | LEADING | TRAILING

(Facultatif) Spécifie l'emplacement à partir duquel les caractères doivent être tronqués. Utilisez BOTH pour supprimer les caractères de début et de fin, utilisez LEADING pour supprimer uniquement les caractères de début et utilisez TRAILING pour supprimer uniquement les caractères de fin. Si ce paramètre est omis, les caractères de début et de fin sont tronqués.

trim_chars

(Facultatif) Caractères à tronquer à partir de la chaîne. Si ce paramètre est oublié, les blancs sont tronqués.

string

Chaîne à tronquer.

Type de retour

La fonction TRIM renvoie une chaîne VARCHAR ou CHAR. Si vous utilisez la fonction TRIM avec une commande SQL, Amazon Redshift convertit implicitement les résultats en VARCHAR. Si vous utilisez la fonction TRIM dans la liste SELECT dune fonction SQL et qu'Amazon Redshift ne convertit pas implicitement les résultats, vous devrez peut-être effectuer une conversion explicite pour éviter une erreur de décalage de type de données. Pour plus d'informations sur les conversions explicites, consultez les fonctions Fonction CAST et Fonction CONVERT.

Exemples

Pour tronquer les espaces de début et de fin de la chaîne dog , utilisez l'exemple suivant.

```
SELECT TRIM('    dog    ');

+-------+
| btrim |
+-------+
| dog   |
+-------+
```

Pour tronquer les espaces de début et de fin de la chaîne dog , utilisez l'exemple suivant.

```
SELECT TRIM(BOTH FROM '    dog    ');
```

```
+-------+
| btrim |
+-------+
| dog   |
+-------+
```

Pour supprimer les guillemets doubles ouvrants de la chaîne "dog", utilisez l'exemple suivant.

```
SELECT TRIM(LEADING '"' FROM'"dog"');

+-------+
| ltrim |
+-------+
| dog"  |
+-------+
```

Pour supprimer les guillemets doubles fermants de la chaîne "dog", utilisez l'exemple suivant.

```
SELECT TRIM(TRAILING '"' FROM'"dog"');

+-------+
| rtrim |
+-------+
| "dog  |
+-------+
```

TRIM supprime les caractères de trim_chars qui apparaissent au début ou à la fin de l'argument string. L'exemple suivant tronque les caractères « C », « D » et « G » lorsqu'ils figurent au début ou à la fin de VENUENAME, qui est une colonne VARCHAR. Pour de plus amples informations, veuillez consulter Exemple de base de données.

```
SELECT venueid, venuename, TRIM('CDG' FROM venuename)
FROM venue
WHERE venuename LIKE '%Park'
ORDER BY 2
LIMIT 7;

+---------+----------------------------+----------------------------+
| venueid |         venuename          |           btrim            |
+---------+----------------------------+----------------------------+
|     121 | AT&T Park                  | AT&T Park                  |
|     109 | Citizens Bank Park         | itizens Bank Park          |
```

```
|     102 | Comerica Park              | omerica Park              |
|       9 | Dick's Sporting Goods Park | ick's Sporting Goods Park |
|      97 | Fenway Park                | Fenway Park               |
|     112 | Great American Ball Park   | reat American Ball Park   |
|     114 | Miller Park                | Miller Park               |
+---------+----------------------------+---------------------------+
```

Fonction UPPER

Convertit la valeur en majuscules. UPPER prend en charge les caractères à plusieurs octets UTF-8, à concurrence de quatre octets au maximum par caractère.

Syntaxe

```
UPPER(string)
```

Arguments

string

Le paramètre d'entrée est une chaîne VARCHAR ou tout autre type de données, tel que CHAR, qui peut être implicitement converti en VARCHAR.

Type de retour

La fonction UPPER renvoie une chaîne de caractères qui est du même type que la chaîne d'entrée. Par exemple, la fonction renvoie une chaîne VARCHAR si l'entrée est une chaîne VARCHAR.

Exemples

L'exemple suivant utilise les données de la table CATEGORY de l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

Pour convertir le champ CATNAME en majuscules, utilisez ce qui suit.

```
SELECT catname, UPPER(catname)
FROM category
ORDER BY 1,2;

+-----------+-----------+
|  catname  |   upper   |
+-----------+-----------+
```

```
| Classical | CLASSICAL |
| Jazz      | JAZZ      |
| MLB       | MLB       |
| MLS       | MLS       |
| Musicals  | MUSICALS  |
| NBA       | NBA       |
| NFL       | NFL       |
| NHL       | NHL       |
| Opera     | OPERA     |
| Plays     | PLAYS     |
| Pop       | POP       |
+-----------+-----------+
```

Fonctions d'informations sur le type SUPER

Vous trouverez ci-dessous une description des fonctions d'informations de type pour SQL prises en charge par Amazon Redshift pour dériver les informations dynamiques à partir des entrées du type de données SUPER.

Rubriques

- Fonction DECIMAL_PRECISION
- Fonction DECIMAL_SCALE
- Fonction IS_ARRAY
- Fonction IS_BIGINT
- Fonction IS_BOOLEAN
- Fonction IS_CHAR
- Fonction IS_DECIMAL
- Fonction IS_FLOAT
- Fonction IS_INTEGER
- Fonction IS_OBJECT
- Fonction IS_SCALAR
- Fonction IS_SMALLINT
- Fonction IS_VARCHAR
- Fonction JSON_SIZE
- Fonction JSON_TYPEOF
- SIZE

Fonction DECIMAL_PRECISION

Vérifie la précision du nombre total maximal de chiffres décimaux à stocker. Ce nombre comprend les chiffres qui se situent à gauche et à droite de la virgule décimale. La plage de précision va de 1 à 38, avec une valeur par défaut de 38.

Syntaxe

```
DECIMAL_PRECISION(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

INTEGER

Exemples

Pour appliquer la fonction DECIMAL_PRECISION à la table t, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (3.14159);

SELECT DECIMAL_PRECISION(s) FROM t;

+-------------------+
| decimal_precision |
+-------------------+
|                 6 |
+-------------------+
```

Fonction DECIMAL_SCALE

Vérifie le nombre de chiffres décimaux à stocker à droite de la virgule. La plage de l'échelle est comprise entre 0 et le point de précision, avec une valeur par défaut de 0.

Syntaxe

```
DECIMAL_SCALE(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

INTEGER

Exemples

Pour appliquer la fonction DECIMAL_SCALE à la table t, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (3.14159);

SELECT DECIMAL_SCALE(s) FROM t;

+---------------+
| decimal_scale |
+---------------+
|             5 |
+---------------+
```

Fonction IS_ARRAY

Vérifie si une variable est un tableau. La fonction renvoie `true` si la variable est un tableau. La fonction inclut également les tableaux vides. Sinon, la fonction renvoie `false` pour toutes les autres valeurs, y compris null.

Syntaxe

```
IS_ARRAY(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

BOOLEAN

Exemples

Pour vérifier si `[1,2]` est un tableau à l'aide de la fonction IS_ARRAY, utilisez l'exemple suivant.

```
SELECT IS_ARRAY(JSON_PARSE('[1,2]'));

+----------+
| is_array |
+----------+
| true     |
+----------+
```

Fonction IS_BIGINT

Vérifie si une valeur est de type `BIGINT`. La fonction IS_BIGINT renvoie `true` pour les nombres d'échelle 0 dans la plage de 64 bits. Sinon, la fonction renvoie `false` pour toutes les autres valeurs, y compris null et les nombres à virgule flottante.

La fonction IS_BIGINT est un sur-ensemble de IS_INTEGER.

Syntaxe

```
IS_BIGINT(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

BOOLEAN

Exemples

Pour vérifier si 5 est de type BIGINT à l'aide de la fonction IS_BIGINT, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (5);

SELECT s, IS_BIGINT(s) FROM t;

+---+-----------+
| s | is_bigint |
+---+-----------+
| 5 | true      |
+---+-----------+
```

Fonction IS_BOOLEAN

Vérifie si une valeur est de type BOOLEAN. La fonction IS_BOOLEAN renvoie `true` pour les booléens JSON constants. La fonction renvoie `false` pour toutes les autres valeurs, y compris null.

Syntaxe

```
IS_BOOLEAN(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

BOOLEAN

Exemples

Pour vérifier si TRUE est de type BOOLEAN à l'aide de la fonction IS_BOOLEAN, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (TRUE);

SELECT s, IS_BOOLEAN(s) FROM t;

+------+------------+
|  s   | is_boolean |
+------+------------+
| true | true       |
+------+------------+
```

Fonction IS_CHAR

Vérifie si une valeur est de type CHAR. La fonction IS_CHAR renvoie `true` pour les chaînes qui contiennent uniquement des caractères ASCII, car le type CHAR ne peut stocker que des caractères au format ASCII. La fonction renvoie `false` pour toutes les autres valeurs.

Syntaxe

```
IS_CHAR(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

BOOLEAN

Exemples

Pour vérifier si `t` est de type CHAR à l'aide de la fonction IS_CHAR, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES ('t');
```

```
SELECT s, IS_CHAR(s) FROM t;

+-----+---------+
|  s  | is_char |
+-----+---------+
| "t" | true    |
+-----+---------+
```

Fonction IS_DECIMAL

Vérifie si une valeur est de type `DECIMAL`. La fonction IS_DECIMAL renvoie `true` pour les nombres qui ne sont pas à virgule flottante. La fonction renvoie `false` pour toutes les autres valeurs, y compris null.

La fonction IS_DECIMAL est un sur-ensemble de IS_BIGINT.

Syntaxe

```
IS_DECIMAL(super_expression)
```

Arguments

super_expression

Expression ou colonne `SUPER`.

Type de retour

`BOOLEAN`

Exemples

Pour vérifier si `1.22` est de type `DECIMAL` à l'aide de la fonction IS_DECIMAL, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (1.22);

SELECT s, IS_DECIMAL(s) FROM t;
```

```
+------+------------+
|  s   | is_decimal |
+------+------------+
| 1.22 | true       |
+------+------------+
```

Fonction IS_FLOAT

Vérifie si une valeur est un nombre à virgule flottante. La fonction IS_FLOAT renvoie `true` pour les nombres à virgule flottante (`FLOAT4` et `FLOAT8`). La fonction renvoie `false` pour toutes les autres valeurs.

L'ensemble IS_DECIMAL et l'ensemble IS_FLOAT sont dissociés.

Syntaxe

```
IS_FLOAT(super_expression)
```

Arguments

super_expression

Expression ou colonne `SUPER`.

Type de retour

`BOOLEAN`

Exemples

Pour vérifier si `2.22::FLOAT` est de type `FLOAT` à l'aide de la fonction IS_FLOAT, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES(2.22::FLOAT);

SELECT s, IS_FLOAT(s) FROM t;

+---------+----------+
|    s    | is_float |
```

```
+---------+----------+
| 2.22e+0 | true     |
+---------+----------+
```

Fonction IS_INTEGER

Renvoie `true` pour les nombres d'échelle 0 dans la plage de 32 bits, et `false` pour toutes les autres valeurs (y compris null et les nombres à virgule flottante).

La fonction IS_INTEGER est un sur-ensemble de la fonction IS_SMALLINT.

Syntaxe

```
IS_INTEGER(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

BOOLEAN

Exemples

Pour vérifier si 5 est de type INTEGER à l'aide de la fonction IS_INTEGER, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (5);

SELECT s, IS_INTEGER(s) FROM t;

+---+------------+
| s | is_integer |
+---+------------+
| 5 | true       |
+---+------------+
```

Fonction IS_OBJECT

Vérifie si une variable est un objet. La fonction IS_OBJECT renvoie `true` pour les objets, y compris les objets vides. La fonction renvoie `false` pour toutes les autres valeurs, y compris null.

Syntaxe

```
IS_OBJECT(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

BOOLEAN

Exemples

Pour vérifier si `{"name": "Joe"}` est un objet à l'aide de la fonction IS_OBJECT, utilisez l'exemple suivant.

```
CREATE TABLE t(s super);

INSERT INTO t VALUES (JSON_PARSE('{"name": "Joe"}'));

SELECT s, IS_OBJECT(s) FROM t;

+----------------+-----------+
|       s        | is_object |
+----------------+-----------+
| {"name":"Joe"} | true      |
+----------------+-----------+
```

Fonction IS_SCALAR

Vérifie si une variable est un scalaire. La fonction IS_SCALAR renvoie `true` pour toute valeur qui n'est pas un tableau ou un objet. La fonction renvoie `false` pour toutes les autres valeurs, y compris null.

L'ensemble IS_ARRAY, IS_OBJECT et IS_SCALAR couvre toutes les valeurs à l'exception des valeurs null.

Syntaxe

```
IS_SCALAR(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

BOOLEAN

Exemples

Pour vérifier si `{"name": "Joe"}` est un scalaire à l'aide de la fonction IS_SCALAR, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (JSON_PARSE('{"name": "Joe"}'));

SELECT s, IS_SCALAR(s.name) FROM t;

+----------------+-----------+
|       s        | is_scalar |
+----------------+-----------+
| {"name":"Joe"} | true      |
+----------------+-----------+
```

Fonction IS_SMALLINT

Vérifie si une variable est de type SMALLINT. La fonction IS_SMALLINT renvoie `true` pour les nombres d'échelle 0 dans la plage de 16 bits. La fonction renvoie `false` pour toutes les autres valeurs, y compris null et les nombres à virgule flottante.

Syntaxe

```
IS_SMALLINT(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Return

BOOLEAN

Exemples

Pour vérifier si 5 est de type SMALLINT à l'aide de la fonction IS_SMALLINT, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES (5);

SELECT s, IS_SMALLINT(s) FROM t;

+---+-------------+
| s | is_smallint |
+---+-------------+
| 5 | true        |
+---+-------------+
```

Fonction IS_VARCHAR

Vérifie si une variable est de type VARCHAR. La fonction IS_VARCHAR renvoie `true` pour toutes les chaînes. La fonction renvoie `false` pour toutes les autres valeurs.

La fonction IS_VARCHAR est un sur-ensemble de la fonction IS_CHAR.

Syntaxe

```
IS_VARCHAR(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

BOOLEAN

Exemples

Pour vérifier si abc est de type VARCHAR à l'aide de la fonction IS_VARCHAR, utilisez l'exemple suivant.

```
CREATE TABLE t(s SUPER);

INSERT INTO t VALUES ('abc');

SELECT s, IS_VARCHAR(s) FROM t;

+-------+------------+
|   s   | is_varchar |
+-------+------------+
| "abc" | true       |
+-------+------------+
```

Fonction JSON_SIZE

La fonction JSON_SIZE renvoie le nombre d'octets présents dans l'expression SUPER donnée lorsqu'elle est sérialisée en chaîne.

Syntaxe

```
JSON_SIZE(super_expression)
```

Arguments

super_expression

Constante ou expression SUPER.

Type de retour

`INTEGER`

La fonction JSON_SIZE renvoie un `INTEGER` indiquant le nombre d'octets présents dans la chaîne en entrée. Cette valeur est différente du nombre de caractères. Par exemple, le caractère UTF-8 #, un point noir, a une taille de 3 octets bien qu'il s'agisse d'un seul caractère.

Notes d'utilisation

JSON_SIZE(x) est fonctionnellement identique à OCTET_LENGTH(JSON_SERIALIZE). Notez toutefois que JSON_SERIALIZE renvoie une erreur quand l'expression `SUPER` fournie dépasse la limite `VARCHAR` du système lorsqu'elle est sérialisée. JSON_SIZE ne présente pas cette limitation.

Exemples

Pour renvoyer la longueur d'une valeur `SUPER` sérialisée en chaîne, utilisez l'exemple suivant.

```
SELECT JSON_SIZE(JSON_PARSE('[10001,10002,"#"]'));

+-----------+
| json_size |
+-----------+
|        19 |
+-----------+
```

Notez que l'expression `SUPER` fournie comporte 17 caractères, mais que # est un caractère de 3 octets, si bien que JSON_SIZE renvoie 19.

Fonction JSON_TYPEOF

La fonction scalaire JSON_TYPEOF renvoie un `VARCHAR` avec les valeurs boolean, number, string, object, array ou null, selon le type dynamique de la valeur `SUPER`.

Syntaxe

```
JSON_TYPEOF(super_expression)
```

Arguments

super_expression

Expression ou colonne SUPER.

Type de retour

VARCHAR

Exemples

Pour vérifier le type de JSON pour le tableau [1,2] à l'aide de la fonction JSON_TYPEOF, utilisez l'exemple suivant.

```
SELECT JSON_TYPEOF(ARRAY(1,2));

+-------------+
| json_typeof |
+-------------+
| array       |
+-------------+
```

Pour vérifier le type de JSON pour l'objet {"name":"Joe"} à l'aide de la fonction JSON_TYPEOF, utilisez l'exemple suivant.

```
SELECT JSON_TYPEOF(JSON_PARSE('{"name":"Joe"}'));

+-------------+
| json_typeof |
+-------------+
| object      |
+-------------+
```

SIZE

Renvoie la taille binaire en mémoire d'une constante ou d'une expression de type SUPER sous la forme d'un INTEGER.

Syntaxe

```
SIZE(super_expression)
```

Arguments

super_expression

Constante ou expression de type SUPER.

Type de retour

INTEGER

Exemples

Pour utiliser SIZE pour obtenir la taille en mémoire de plusieurs expressions de type SUPER, utilisez l'exemple suivant.

```
CREATE TABLE test_super_size(a SUPER);

INSERT INTO test_super_size
VALUES
  (null),
  (TRUE),
  (JSON_PARSE('[0,1,2,3]')),
  (JSON_PARSE('{"a":0,"b":1,"c":2,"d":3}'))
;

SELECT a, SIZE(a)
FROM test_super_size
ORDER BY 2, 1;

+---------------------------+------+
|             a             | size |
+---------------------------+------+
| true                      |    4 |
| NULL                      |    4 |
| [0,1,2,3]                 |   23 |
| {"a":0,"b":1,"c":2,"d":3} |   52 |
+---------------------------+------+
```

Fonctions et opérateurs VARBYTE

Les fonctions et opérateurs Amazon Redshift prenant en charge le type de données VARBYTE incluent :

- Opérateurs VARBYTE
- FROM_HEX
- FROM_VARBYTE
- GETBIT
- TO_HEX
- TO_VARBYTE
- CONCAT
- LEN
- Fonction LENGTH
- OCTET_LENGTH
- Fonction SUBSTRING

Opérateurs VARBYTE

Le tableau suivant répertorie les opérateurs VARBYTE. L'opérateur fonctionne avec une valeur binaire de type de données VARBYTE. Si une entrée ou les deux ont une valeur null, le résultat a une valeur null.

Opérateurs pris en charge

Opérateur	Description	Type de retour
<	Inférieur à	BOOLEAN
<=	Inférieur ou égal à	BOOLEAN
=	Égal à	BOOLEAN

Opérateur	Description	Type de retour
>	Supérieure à	BOOLEAN
>=	Supérieur ou égal à	BOOLEAN
!= ou <>	Non égal à	BOOLEAN
||	Concaténation	VARBYTE
+	Concaténation	VARBYTE
~	Au niveau du bit not	VARBYTE
&	Au niveau du bit et	VARBYTE
|	Au niveau du bit or	VARBYTE
#	Au niveau du bit xor	VARBYTE

Exemples

Dans les exemples suivants, la valeur de `'a'::VARBYTE` est 61 et la valeur de `'b'::VARBYTE` est 62. L'élément `::` convertit les chaînes en type de données VARBYTE. Pour plus d'informations sur la conversion des types de données, consultez CAST.

Pour déterminer si `'a'` est inférieur à `'b'` à l'aide de l'opérateur <, utilisez l'exemple suivant.

```
SELECT 'a'::VARBYTE < 'b'::VARBYTE AS less_than;

+-----------+
```

```
| less_than |
+-----------+
| true      |
+-----------+
```

Pour déterminer si `'a'` est égal à `'b'` à l'aide de l'opérateur =, utilisez l'exemple suivant.

```
SELECT 'a'::VARBYTE = 'b'::VARBYTE AS equal;

+-------+
| equal |
+-------+
| false |
+-------+
```

Pour concaténer deux valeurs binaires à l'aide de l'opérateur `||`, utilisez l'exemple suivant.

```
SELECT 'a'::VARBYTE || 'b'::VARBYTE AS concat;

+--------+
| concat |
+--------+
|   6162 |
+--------+
```

Pour concaténer deux valeurs binaires à l'aide de l'opérateur +, utilisez l'exemple suivant.

```
SELECT 'a'::VARBYTE + 'b'::VARBYTE AS concat;

+--------+
| concat |
+--------+
|   6162 |
+--------+
```

Pour annuler chaque bit de la valeur binaire d'entrée à l'aide de la fonction FROM_VARBYTE, utilisez l'exemple suivant. La chaîne `'a'` a pour valeur `01100001`. Pour de plus amples informations, veuillez consulter FROM_VARBYTE.

```
SELECT FROM_VARBYTE(~'a'::VARBYTE, 'binary');
```

```
+--------------+
| from_varbyte |
+--------------+
|     10011110 |
+--------------+
```

Pour appliquer l'opérateur & sur les deux valeurs binaires d'entrée, utilisez l'exemple suivant. La chaîne `'a'` a pour valeur `01100001` et `'b'` a pour valeur `01100010`.

```
SELECT FROM_VARBYTE('a'::VARBYTE & 'b'::VARBYTE, 'binary');

+--------------+
| from_varbyte |
+--------------+
|     01100000 |
+--------------+
```

Fonction FROM_HEX

FROM_HEX convertit une valeur hexadécimale en valeur binaire.

Syntaxe

```
FROM_HEX(hex_string)
```

Arguments

hex_string

Chaîne hexadécimale de type de données `VARCHAR` ou `TEXT` à convertir. Le format doit être une valeur littérale.

Type de retour

`VARBYTE`

Exemples

Pour convertir la représentation hexadécimale de `'6162'` en une valeur binaire, utilisez l'exemple suivant. Le résultat est automatiquement affiché sous forme de représentation hexadécimale de la valeur binaire.

```
SELECT FROM_HEX('6162');

+----------+
| from_hex |
+----------+
|     6162 |
+----------+
```

Fonction FROM_VARBYTE

FROM_VARBYTE convertit une valeur binaire en chaîne de caractères au format spécifié.

Syntaxe

```
FROM_VARBYTE(binary_value, format)
```

Arguments

binary_value

Valeur binaire du type de données VARBYTE.

format

Format de la chaîne de caractères renvoyée. Les valeurs valides insensibles à la casse sont `hex`, `binary`, `utf8` (également `utf-8` et `utf_8`) et `base64`.

Type de retour

VARCHAR

Exemples

Pour convertir la valeur binaire `'ab'` en une valeur hexadécimale, utilisez l'exemple suivant.

```
SELECT FROM_VARBYTE('ab', 'hex');

+--------------+
| from_varbyte |
+--------------+
|         6162 |
+--------------+
```

Pour renvoyer la représentation binaire de `'4d'`, utilisez l'exemple suivant. La représentation binaire de `'4d'` est la chaîne de caractères `01001101`.

```
SELECT FROM_VARBYTE(FROM_HEX('4d'), 'binary');

+--------------+
| from_varbyte |
+--------------+
|     01001101 |
+--------------+
```

Fonction GETBIT

GETBIT renvoie la valeur de bit d'une valeur binaire à l'index spécifié.

Syntaxe

```
GETBIT(binary_value, index)
```

Arguments

binary_value

Valeur binaire du type de données VARBYTE.

index

Numéro d'index du bit dans la valeur binaire renvoyée. La valeur binaire est un tableau de bits basé sur 0 qui est indexé du bit le plus à droite (bit le moins significatif) au bit le plus à gauche (bit le plus significatif).

Type de retour

`INTEGER`

Exemples

Pour renvoyer le bit à l'index 2 de la valeur binaire `from_hex('4d')`, utilisez l'exemple suivant. La représentation binaire de `'4d'` est `01001101`.

```
SELECT GETBIT(FROM_HEX('4d'), 2);
```

```
+--------+
| getbit |
+--------+
|      1 |
+--------+
```

Pour renvoyer les bits aux huit emplacements d'index de la valeur binaire renvoyée par `from_hex('4d')`, utilisez l'exemple suivant. La représentation binaire de `'4d'` est `01001101`.

```
SELECT GETBIT(FROM_HEX('4d'), 7), GETBIT(FROM_HEX('4d'), 6),
  GETBIT(FROM_HEX('4d'), 5), GETBIT(FROM_HEX('4d'), 4),
  GETBIT(FROM_HEX('4d'), 3), GETBIT(FROM_HEX('4d'), 2),
  GETBIT(FROM_HEX('4d'), 1), GETBIT(FROM_HEX('4d'), 0);

+--------+--------+--------+--------+--------+--------+--------+--------+
| getbit | getbit | getbit | getbit | getbit | getbit | getbit | getbit |
+--------+--------+--------+--------+--------+--------+--------+--------+
|      0 |      1 |      0 |      0 |      1 |      1 |      0 |      1 |
+--------+--------+--------+--------+--------+--------+--------+--------+
```

Fonction TO_HEX

TO_HEX convertit un nombre ou une valeur binaire en une représentation hexadécimale.

Syntaxe

```
TO_HEX(value)
```

Arguments

valeur

Nombre ou valeur binaire (VARBYTE) à convertir.

Type de retour

VARCHAR

Exemples

Pour convertir un nombre en sa représentation hexadécimale, utilisez l'exemple suivant.

```
SELECT TO_HEX(2147676847);

+----------+
|  to_hex  |
+----------+
| 8002f2af |
+----------+
```

Pour convertir la représentation VARBYTE de `'abc'` en un nombre hexadécimal, utilisez l'exemple suivant.

```
SELECT TO_HEX('abc'::VARBYTE);

+--------+
| to_hex |
+--------+
| 616263 |
+--------+
```

Pour créer une table, insérer la représentation VARBYTE de `'abc'` dans un nombre hexadécimal et sélectionner la colonne avec cette valeur, utilisez l'exemple suivant.

```
CREATE TABLE t (vc VARCHAR);
INSERT INTO t SELECT TO_HEX('abc'::VARBYTE);
SELECT vc FROM t;

+--------+
|   vc   |
+--------+
| 616263 |
+--------+
```

Pour montrer que, lors de la conversion d'une valeur VARBYTE en VARCHAR, le format est UTF-8, utilisez l'exemple suivant.

```
CREATE TABLE t (vc VARCHAR);
INSERT INTO t SELECT 'abc'::VARBYTE::VARCHAR;

SELECT vc FROM t;

+-----+
```

```
| vc  |
+-----+
| abc |
+-----+
```

Fonction TO_VARBYTE

TO_VARBYTE convertit une chaîne dans un format spécifié en valeur binaire.

Syntaxe

```
TO_VARBYTE(string, format)
```

Arguments

string

Chaîne CHAR ou VARCHAR.

format

Format du fichier d'entrée. Les valeurs valides insensibles à la casse sont `hex`, `binary`, `utf8` (également `utf-8` et `utf_8`) et `base64`.

Type de retour

VARBYTE

Exemples

Pour convertir la valeur hexadécimale 6162 en une valeur binaire, utilisez l'exemple suivant. Le résultat est automatiquement affiché sous forme de représentation hexadécimale de la valeur binaire.

```
SELECT TO_VARBYTE('6162', 'hex');

+------------+
| to_varbyte |
+------------+
|       6162 |
+------------+
```

Pour renvoyer la représentation binaire de 4d, utilisez l'exemple suivant. La représentation binaire de « 4d » est `01001101`.

```
SELECT TO_VARBYTE('01001101', 'binary');

+------------+
| to_varbyte |
+------------+
|         4d |
+------------+
```

Pour convertir la chaîne `'a'` dans UTF-8 en une valeur binaire, utilisez l'exemple suivant. Le résultat est automatiquement affiché sous forme de représentation hexadécimale de la valeur binaire.

```
SELECT TO_VARBYTE('a', 'utf8');

+------------+
| to_varbyte |
+------------+
|         61 |
+------------+
```

Pour convertir la chaîne `'4'` hexadécimale en une valeur binaire, utilisez l'exemple suivant. Si la longueur de la chaîne hexadécimale est un nombre impair, alors un `0` est ajouté pour former un nombre hexadécimal valide.

```
SELECT TO_VARBYTE('4', 'hex');

+------------+
| to_varbyte |
+------------+
|         04 |
+------------+
```

Fonctions de fenêtrage

En utilisant les fonctions de fenêtrage, vous pouvez créer des requêtes d’analyse commerciale plus efficacement. Les fonctions de fenêtrage fonctionnent sur une partition ou « fenêtre » d’un ensemble de résultats et renvoient une valeur pour chaque ligne de cette fenêtre. En revanche, les fonctions non fenêtrées effectuent leurs calculs sur chaque ligne du jeu de résultats. Contrairement aux fonctions de groupe qui regroupent les lignes de résultats, les fonctions de fenêtrage conservent toutes les lignes de l’expression de table.

Les valeurs renvoyées sont calculées en utilisant les valeurs des ensembles de lignes de cette fenêtre. Pour chaque ligne de la table, la fenêtre définit un ensemble de lignes qui est utilisé pour calculer des attributs supplémentaires. Une fenêtre est définie à l'aide d'une spécification de fenêtrage (clause OVER) et s'appuie sur trois concepts principaux :

- Le partitionnement de fenêtrage qui constitue des groupes de lignes (clause PARTITION)
- L'ordonnancement de fenêtrage, qui définit un ordre ou une séquence de lignes dans chaque partition (clause ORDER BY)
- Les cadres de fenêtrage, qui sont définis par rapport à chaque ligne afin de limiter davantage l'ensemble de lignes (spécification ROWS)

Les fonctions de fenêtrage constituent le dernier ensemble d'opérations effectuées dans une requête à l'exception de la clause ORDER BY finale. Toutes les jointures et toutes les clauses WHERE, GROUP BY et HAVING doivent être terminées avant que les fonctions de fenêtrage soient traitées. Par conséquent, les fonctions de fenêtrage peuvent s'afficher uniquement dans la liste de sélection ou la clause ORDER BY. Vous pouvez utiliser plusieurs fonctions de fenêtrage dans une seule requête avec différentes clauses de cadre. Vous pouvez également utiliser des fonctions de fenêtrage dans d'autres expressions scalaires, telles que CASE.

Les fonctions de fenêtre ne peuvent pas être imbriquées. Par exemple, une fonction d'agrégation SUM peut apparaître dans une fonction de fenêtreSUM, mais une fonction de fenêtre SUM ne peut pas apparaître dans une autre fonction de fenêtre SUM. Ce qui suit n'est pas pris en charge car une fonction de fenêtre est imbriquée dans une autre fonction de fenêtre.

```
SELECT SUM(SUM(selectcol) OVER (PARTITION BY ordercol)) OVER (Partition by ordercol)
 FROM t;
```

Récapitulatif de la syntaxe de la fonction de fenêtrage

Les fonctions de fenêtre suivent la syntaxe standard suivante.

```
function (expression) OVER (
[ PARTITION BY expr_list ]
[ ORDER BY order_list [ frame_clause ] ] )
```

Ici, function est l'une des fonctions décrites dans cette section.

L'expr_list se présente comme suit.

```
expression | column_name [, expr_list ]
```

L'order_list se présente comme suit.

```
expression | column_name [ ASC | DESC ]
[ NULLS FIRST | NULLS LAST ]
[, order_list ]
```

La frame_clause se présente comme suit.

```
ROWS
{ UNBOUNDED PRECEDING | unsigned_value PRECEDING | CURRENT ROW } |

{ BETWEEN
{ UNBOUNDED PRECEDING | unsigned_value { PRECEDING | FOLLOWING } | CURRENT ROW}
AND
{ UNBOUNDED FOLLOWING | unsigned_value { PRECEDING | FOLLOWING } | CURRENT ROW }}
```

Arguments

fonction

La fonction de fenêtrage. Pour plus d'informations, consultez les descriptions de chaque fonction.

OVER

La clause qui définit la spécification du fenêtrage. La clause OVER est obligatoire pour les fonctions de fenêtrage et différencie les fonctions de fenêtrage d'autres fonctions SQL.

PARTITION BY expr_list

(Facultatif) La clause PARTITION BY subdivise le jeu de résultats en partitions, comme la clause GROUP BY. Si une clause de partition est présente, la fonction est calculée pour les lignes de chaque partition. Si aucune clause de partition n'est spécifiée, une seule partition contient la totalité de la table et la fonction est calculée pour cette table complète.

Les fonctions de rang DENSE_RANK, NTILE, RANK et ROW_NUMBER, nécessitent une comparaison globale de toutes les lignes du jeu de résultats. Lorsqu'une clause PARTITION BY est utilisée, l'optimiseur de requête peut exécuter chaque agrégation en parallèle en répartissant la charge de travail sur plusieurs tranches selon les partitions. Si la clause PARTITION BY n'est pas présente, l'étape d'agrégation doit être exécutée en série sur une seule tranche, ce qui

peut avoir une incidence négative importante sur les performances, surtout pour des clusters de grande taille.

Amazon Redshift ne prend pas en charge les littéraux de chaîne dans les clauses PARTITION BY.

ORDER BY order_list

(Facultatif) La fonction de fenêtrage est appliquée aux lignes de chaque partition triées selon la spécification d'ordre de ORDER BY. Cette clause ORDER BY est distincte et sans aucun lien avec une clause ORDER BY dans la frame_clause. La clause ORDER BY peut être utilisée sans la clause PARTITION BY.

Pour les fonctions de rang, la clause ORDER BY identifie les mesures des valeurs de rang. Pour les fonctions d'agrégation, les lignes partitionnées doivent être ordonnées avant que la fonction d'agrégation soit calculée pour chaque cadre. Pour en savoir plus sur les types de fonction de fenêtrage, consultez Fonctions de fenêtrage.

Les identificateurs de colonnes ou les expressions qui correspondent aux identificateurs de colonnes sont requis dans la liste d'ordre. Ni les constantes, ni les expressions constantes ne peuvent être utilisées pour remplacer les noms de colonnes.

Les valeurs NULLS sont traitées comme leur propre groupe, triées et classées selon l'option NULLS FIRST ou NULLS LAST. Par défaut, les valeurs NULL sont triées et classées en dernier par ordre croissant (ASC) et triées et classées en premier par ordre décroissant (DESC).

Amazon Redshift ne prend pas en charge les littéraux de chaîne dans les clauses ORDER BY.

Si la clause ORDER BY est omise, l'ordre des lignes est non déterministe.

> **Note**
>
> Dans un système parallèle tel qu'Amazon Redshift, quand une clause ORDER BY ne génère pas d'ordonnancement unique et total des données, l'ordre des lignes est non déterministe. Autrement dit, si l'expression ORDER BY produit des valeurs en double (ordonnancement partiel), l'ordre de renvoi de ces lignes peut varier d'une exécution d'Amazon Redshift à l'autre. De leur côté, les fonctions de fenêtrage peuvent renvoyer des résultats inattendus ou incohérents. Pour plus d'informations, consultez Ordonnancement unique des données pour les fonctions de fenêtrage.

column_name

Nom d’une colonne à partitionner ou à ordonner.

ASC | DESC

Option qui définit l’ordre de tri de l’expression, comme suit :

- ASC : croissant (par exemple, de faible à élevé pour les valeurs numériques et de « A » à « Z » pour les chaînes de caractères). Si aucune option n’est spécifiée, les données sont triées dans l’ordre croissant par défaut.
- DESC : descendantes (valeurs d’élevées à faibles pour les valeurs numériques ; de « Z » à « A » pour les chaînes).

NULLS FIRST | NULLS LAST

Option qui spécifie si les valeurs NULLS devraient être classés en premier, avant les valeurs non NULL, ou en dernier, après les valeurs non NULL. Par défaut, les valeurs NULLS sont triées et classées en dernier par ordre croissant (ASC) et triées et classées en premier par ordre décroissant (DESC).

frame_clause

Pour les fonctions d’agrégation, la clause de cadre affine l’ensemble de lignes dans la fenêtre d’une fonction lorsque vous utilisez ORDER BY. Elle vous permet d’inclure ou d’exclure des ensembles de lignes dans le résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés.

La clause frame ne s’applique pas aux fonctions de classement. En outre, la clause de cadre n’est pas requise lorsqu’aucune clause ORDER BY n’est utilisée dans la clause OVER pour une fonction d’agrégation. Si une clause ORDER BY est utilisée pour une fonction d’agrégation, une clause de cadre explicite est requise.

Si aucune clause ORDER BY n’est spécifiée, le cadre implicite est sans limite : équivalent à ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING.

ROWS

Cette clause définit le cadre de fenêtrage en spécifiant un décalage physique de la ligne actuelle.

Cette clause spécifie les lignes de la fenêtre ou de la partition actuelle auxquelles la valeur de la ligne actuelle doit être associée. Elle utilise des arguments qui spécifient la position de la ligne, qui peut être avant ou après la ligne actuelle. Le point de référence de tous les cadres de fenêtrage

est la ligne actuelle. Chaque ligne devient la ligne actuelle à son tour à mesure que le cadre de fenêtrage avance dans la partition.

Le cadre peut être un simple ensemble de lignes allant jusqu'à et incluant la ligne actuelle.

```
{UNBOUNDED PRECEDING | offset PRECEDING | CURRENT ROW}
```

Ou il peut s'agir d'un ensemble de lignes situées entre les deux limites.

```
BETWEEN
{ UNBOUNDED PRECEDING | offset { PRECEDING | FOLLOWING } | CURRENT ROW }
AND
{ UNBOUNDED FOLLOWING | offset { PRECEDING | FOLLOWING } | CURRENT ROW }
```

UNBOUNDED PRECEDING indique que la fenêtre commence à la première ligne de la partition ; offset PRECEDING indique que la fenêtre commence un certain nombre de lignes équivalant à la valeur de décalage avant la ligne actuelle. UNBOUNDED PRECEDING est la valeur par défaut.

CURRENT ROW indique que la fenêtre commence ou se termine à la ligne actuelle.

UNBOUNDED FOLLOWING indique que la fenêtre se termine à la dernière ligne de la partition ; offset FOLLOWING indique que la fenêtre se termine un certain nombre de lignes équivalant à la valeur de décalage après la ligne actuelle.

offset identifie un nombre physique de lignes avant ou après la ligne actuelle. Dans ce cas, offset doit être une constante ayant une valeur numérique positive. Par exemple, 5 FOLLOWING arrête les 5 lignes du cadre après la ligne actuelle.

Là où BETWEEN n'est pas spécifié, le cadre est implicitement délimité par la ligne actuelle. Par exemple, `ROWS 5 PRECEDING` est égal à `ROWS BETWEEN 5 PRECEDING AND CURRENT ROW`. En outre, `ROWS UNBOUNDED FOLLOWING` est égal à `ROWS BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING`.

> **Note**
>
> Vous ne pouvez pas spécifier un cadre dans lequel la limite de début est supérieure à la limite de fin. Par exemple, vous ne pouvez pas spécifier l'un des cadres suivants.
>
> ```
> between 5 following and 5 preceding
> between current row and 2 preceding
> ```

```
between 3 following and current row
```

Ordonnancement unique des données pour les fonctions de fenêtrage

Si une clause ORDER BY pour une fonction de fenêtrage ne génère pas d'ordonnancement unique et total des données, l'ordre des lignes est non déterministe. Si l'expression ORDER BY génère des valeurs en double (ordonnancement partiel), l'ordre de ces lignes qui est renvoyé peut varier lors de plusieurs exécutions. Dans ce cas, les fonctions de fenêtrage peuvent également renvoyer des résultats inattendus ou incohérents.

Par exemple, la requête suivante renvoie des résultats différents sur plusieurs exécutions. Ces différents résultats se produisent parce que `order by dateid` ne produit pas d'ordonnancement unique des données pour la fonction de fenêtrage SUM.

```
select dateid, pricepaid,
sum(pricepaid) over(order by dateid rows unbounded preceding) as sumpaid
from sales
group by dateid, pricepaid;

dateid | pricepaid |   sumpaid
--------+-----------+-------------
1827 |   1730.00 |     1730.00
1827 |    708.00 |     2438.00
1827 |    234.00 |     2672.00
...

select dateid, pricepaid,
sum(pricepaid) over(order by dateid rows unbounded preceding) as sumpaid
from sales
group by dateid, pricepaid;

dateid | pricepaid |   sumpaid
--------+-----------+-------------
1827 |    234.00 |      234.00
1827 |    472.00 |      706.00
1827 |    347.00 |     1053.00
...
```

Dans ce cas, l'ajout d'une seconde colonne ORDER BY à la fonction de fenêtrage peut permettre de résoudre le problème.

```
select dateid, pricepaid,
sum(pricepaid) over(order by dateid, pricepaid rows unbounded preceding) as sumpaid
from sales
group by dateid, pricepaid;

dateid | pricepaid | sumpaid
--------+-----------+---------
1827 |    234.00 |  234.00
1827 |    337.00 |  571.00
1827 |    347.00 |  918.00
...
```

Fonctions prises en charge

Amazon Redshift prend en charge deux types de fonctions de fenêtrage : par regroupement et par rang.

Vous trouverez ci-dessous les fonctions d'agrégation prises en charge :

- Fonction de fenêtrage AVG
- Fonction de fenêtrage COUNT
- Fonction de fenêtrage CUME_DIST
- Fonction de fenêtrage DENSE_RANK
- Fonction de fenêtrage FIRST_VALUE
- Fonction de fenêtrage LAG
- Fonction de fenêtrage LAST_VALUE
- Fonction de fenêtrage LEAD
- Fonction de fenêtrage LISTAGG
- Fonction de fenêtrage MAX
- Fonction de fenêtrage MEDIAN
- Fonction de fenêtrage MIN
- Fonction de fenêtrage NTH_VALUE
- Fonction de fenêtrage PERCENTILE_CONT
- Fonction de fenêtrage PERCENTILE_DISC
- Fonction de fenêtrage RATIO_TO_REPORT

- Fonctions de fenêtrage STDDEV_SAMP et STDDEV_POP (STDDEV_SAMP et STDDEV sont synonymes)
- Fonction de fenêtrage SUM
- Fonctions de fenêtrage VAR_SAMP et VAR_POP (VAR_SAMP et VARIANCE sont synonymes)

Vous trouverez ci-dessous les fonctions de classement prises en charge :

- Fonction de fenêtrage DENSE_RANK
- Fonction de fenêtrage NTILE
- Fonction de fenêtrage PERCENT_RANK
- Fonction de fenêtrage RANK
- Fonction de fenêtrage ROW_NUMBER

Exemple de tableau contenant des exemples de fonctions de fenêtrage

Vous trouverez des exemples de fonctions de fenêtrage spécifiques avec la description de chaque fonction. Certains exemples utilisent une table nommée WINSALES, qui contient 11 lignes, tel qu'illustré ci-dessous.

SALESID	DATEID	SELLERID	BUYERID	QTY	QTY_SHIPPED
30001	8/2/2003	3	B	10	10
10001	12/24/2003	1	C	10	10
10005	12/24/2003	1	A	30	
40001	1/9/2004	4	A	40	
10006	1/18/2004	1	C	10	
20001	2/12/2004	2	B	20	20
40005	2/12/2004	4	A	10	10
20002	2/16/2004	2	C	20	20

SALESID	DATEID	SELLERID	BUYERID	QTY	QTY_SHIPPED
30003	4/18/2004	3	B	15	
30004	4/18/2004	3	B	20	
30007	9/7/2004	3	C	30	

Le script suivant crée et remplit l'exemple de table WINSALES.

```
CREATE TABLE winsales(
  salesid int,
  dateid date,
  sellerid int,
  buyerid char(10),
  qty int,
  qty_shipped int);

INSERT INTO winsales VALUES
  (30001, '8/2/2003', 3, 'b', 10, 10),
  (10001, '12/24/2003', 1, 'c', 10, 10),
  (10005, '12/24/2003', 1, 'a', 30, null),
  (40001, '1/9/2004', 4, 'a', 40, null),
  (10006, '1/18/2004', 1, 'c', 10, null),
  (20001, '2/12/2004', 2, 'b', 20, 20),
  (40005, '2/12/2004', 4, 'a', 10, 10),
  (20002, '2/16/2004', 2, 'c', 20, 20),
  (30003, '4/18/2004', 3, 'b', 15, null),
  (30004, '4/18/2004', 3, 'b', 20, null),
  (30007, '9/7/2004', 3, 'c', 30, null);
```

Fonction de fenêtrage AVG

La fonction de fenêtrage AVG renvoie la moyenne (arithmétique) des valeurs d'expression d'entrée. La fonction AVG utilise des valeurs numériques et ignore les valeurs NULL.

Syntaxe

```
AVG ( [ALL ] expression ) OVER
(
```

```
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère.

ALL

Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression pour le compte. La valeur par défaut est ALL. DISTINCT n'est pas pris en charge.

OVER

Spécifie les clauses de fenêtrage des fonctions d'agrégation. La clause OVER différencie les fonctions d'agrégation de fenêtrage des fonctions d'agrégation d'un ensemble normal.

PARTITION BY expr_list

Définit la fenêtre de la fonction AVG en termes d'une ou de plusieurs expressions.

ORDER BY order_list

Trie les lignes dans chaque partition. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY utilise toute la table.

frame_clause

Si une clause ORDER BY est utilisée pour une fonction d'agrégation, une clause de cadre explicite est requise. La clause de cadre affine l'ensemble de lignes dans la fenêtre d'une fonction, en incluant ou en excluant des ensembles de lignes du résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés. Consultez Récapitulatif de la syntaxe de la fonction de fenêtrage.

Types de données

Les types d'argument pris en charge par la fonction AVG sont SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL et DOUBLE PRECISION.

Les types de retour pris en charge par la fonction AVG sont les suivants :

- Arguments BIGINT for SMALLINT ou INTEGER
- Arguments NUMERIC for BIGINT
- DOUBLE PRECISION pour les arguments à virgule flottante

Exemples

L'exemple suivant montre le calcul d'une moyenne mobile des quantités vendues par date, et le classement des résultats par ID de date et ID de vente :

```
select salesid, dateid, sellerid, qty,
avg(qty) over
(order by dateid, salesid rows unbounded preceding) as avg
from winsales
order by 2,1;

salesid |   dateid   | sellerid | qty | avg
---------+------------+----------+-----+-----
30001 | 2003-08-02 |        3 |  10 |  10
10001 | 2003-12-24 |        1 |  10 |  10
10005 | 2003-12-24 |        1 |  30 |  16
40001 | 2004-01-09 |        4 |  40 |  22
10006 | 2004-01-18 |        1 |  10 |  20
20001 | 2004-02-12 |        2 |  20 |  20
40005 | 2004-02-12 |        4 |  10 |  18
20002 | 2004-02-16 |        2 |  20 |  18
30003 | 2004-04-18 |        3 |  15 |  18
30004 | 2004-04-18 |        3 |  20 |  18
30007 | 2004-09-07 |        3 |  30 |  19
(11 rows)
```

Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

Fonction de fenêtrage COUNT

La fonction de fenêtrage COUNT compte les lignes définies par l'expression.

La fonction COUNT se décline en deux variations. COUNT(*) compte toutes les lignes de la table cible, qu'elles comprennent des valeurs null ou non. COUNT(expression) calcule le nombre de lignes avec des valeurs non NULL dans une colonne ou une expression spécifique.

Syntaxe

```
COUNT ( * | [ ALL ] expression) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère.

ALL

Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression pour le compte. La valeur par défaut est ALL. DISTINCT n'est pas pris en charge.

OVER

Spécifie les clauses de fenêtrage des fonctions d'agrégation. La clause OVER différencie les fonctions d'agrégation de fenêtrage des fonctions d'agrégation d'un ensemble normal.

PARTITION BY expr_list

Définit la fenêtre de la fonction COUNT en termes d'une ou de plusieurs expressions.

ORDER BY order_list

Trie les lignes dans chaque partition. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY utilise toute la table.

frame_clause

Si une clause ORDER BY est utilisée pour une fonction d'agrégation, une clause de cadre explicite est requise. La clause de cadre affine l'ensemble de lignes dans la fenêtre d'une fonction, en incluant ou en excluant des ensembles de lignes du résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés. Consultez Récapitulatif de la syntaxe de la fonction de fenêtrage.

Types de données

La fonction COUNT prend en charge tous les types de données d'argument.

Le type de retour pris en charge par la fonction COUNT est BIGINT.

Exemples

L'exemple suivant montre l'affichage de l'ID de ventes, la quantité et le nombre de toutes les lignes dès le début de la fenêtre de données :

```
select salesid, qty,
count(*) over (order by salesid rows unbounded preceding) as count
from winsales
order by salesid;

salesid | qty | count
---------+-----+-----
10001 |  10 |   1
10005 |  30 |   2
10006 |  10 |   3
20001 |  20 |   4
20002 |  20 |   5
30001 |  10 |   6
30003 |  15 |   7
30004 |  20 |   8
30007 |  30 |   9
40001 |  40 |   10
40005 |  10 |   11
(11 rows)
```

Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

L'exemple suivant montre l'affichage de l'ID de ventes, la quantité et le nombre de lignes non null dès le début de la fenêtre de données. (Dans la table WINSALES, la colonne QTY_SHIPPED en contient.) NULLs

```
select salesid, qty, qty_shipped,
count(qty_shipped)
over (order by salesid rows unbounded preceding) as count
from winsales
order by salesid;

salesid | qty | qty_shipped | count
---------+-----+-------------+-------
```

```
10001 |  10 |          10 |   1
10005 |  30 |             |   1
10006 |  10 |             |   1
20001 |  20 |          20 |   2
20002 |  20 |          20 |   3
30001 |  10 |          10 |   4
30003 |  15 |             |   4
30004 |  20 |             |   4
30007 |  30 |             |   4
40001 |  40 |             |   4
40005 |  10 |          10 |   5
(11 rows)
```

Fonction de fenêtrage CUME_DIST

Calcule la distribution cumulée d'une valeur au sein d'une fenêtre ou une partition. En supposant que l'ordre est croissant, la distribution cumulée est déterminée à l'aide de la formule suivante :

```
count of rows with values <= x / count of rows in the window or partition
```

où x est égal à la valeur de la ligne actuelle de la colonne spécifiée dans la clause ORDER BY. Le jeu de données suivant illustre l'utilisation de cette formule :

```
Row# Value   Calculation    CUME_DIST
1        2500    (1)/(5)    0.2
2        2600    (2)/(5)    0.4
3        2800    (3)/(5)    0.6
4        2900    (4)/(5)    0.8
5        3100    (5)/(5)    1.0
```

La plage de valeur de retour est comprise entre >0 et 1, inclus.

Syntaxe

```
CUME_DIST ()
OVER (
[ PARTITION BY partition_expression ]
[ ORDER BY order_list ]
)
```

Arguments

OVER

Clause qui spécifie le partitionnement de fenêtrage. La clause OVER ne peut pas contenir de spécification de cadre de fenêtrage.

PARTITION BY partition_expression

Facultatif. Expression qui définit la plage d'enregistrements de chaque groupe dans la clause OVER.

ORDER BY order_list

Expression permettant de calculer la distribution cumulée. L'expression doit disposer d'un type de données numériques ou être convertible implicitement en une. Si ORDER BY n'est pas spécifié, la valeur de retour est 1 pour toutes les lignes.

Si ORDER BY ne génère pas d'ordonnancement unique, l'ordre des lignes est non déterministe. Pour plus d'informations, consultez Ordonnancement unique des données pour les fonctions de fenêtrage.

Type de retour

FLOAT8

Exemples

L'exemple suivant calcule la distribution cumulée de la quantité par vendeur :

```
select sellerid, qty, cume_dist()
over (partition by sellerid order by qty)
from winsales;

sellerid   qty    cume_dist
-------------------------------------------------
1         10.00    0.33
1         10.64    0.67
1         30.37    1
3         10.04    0.25
3         15.15    0.5
3         20.75    0.75
```

```
3          30.55    1
2          20.09    0.5
2          20.12    1
4          10.12    0.5
4          40.23    1
```

Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

Fonction de fenêtrage DENSE_RANK

La fonction de fenêtrage DENSE_RANK détermine le rang d'une valeur dans un groupe de valeurs, en fonction de l'expression ORDER BY dans la clause OVER. Si la clause PARTITION BY facultative est présente, les rangs sont réinitialisés pour chaque groupe de lignes. Les lignes avec des valeurs égales pour les critères de rang reçoivent le même rang. La fonction DENSE_RANK diffère de RANK sur un point : si deux lignes ou plus sont à égalité, il n'y a pas d'écart dans la séquence des valeurs classées. Par exemple, si deux lignes sont classées 1, le prochain rang est 2.

Vous pouvez avoir des fonctions de rang avec différentes clauses PARTITION BY et ORDER BY dans la même requête.

Syntaxe

```
DENSE_RANK() OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list ]
)
```

Arguments

( )

La fonction ne prend pas d'arguments, mais les parenthèses vides sont obligatoires.

OVER

Clauses de fenêtrage pour la fonction DENSE_RANK.

PARTITION BY expr_list

(Facultatif) Une ou plusieurs expressions qui définissent le fenêtrage.

ORDER BY order_list

(Facultatif) Expression sur laquelle sont basées les valeurs de rang. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY utilise toute la table. Si ORDER BY n'est pas spécifié, la valeur renvoyée est 1 pour toutes les lignes.

Si ORDER BY ne génère pas d'ordonnancement unique, l'ordre des lignes est non déterministe. Pour plus d'informations, consultez Ordonnancement unique des données pour les fonctions de fenêtrage.

Type de retour

BIGINT

Exemples

Les exemples suivants utilisent l'exemple de table pour les fonctions de fenêtrage. Pour de plus amples informations, veuillez consulter Exemple de tableau contenant des exemples de fonctions de fenêtrage.

L'exemple suivant ordonne la table en fonction de la quantité vendue et affecte un rang dense et un rang standard à chaque ligne. Les résultats sont triés une fois que les résultats de la fonction de fenêtrage sont appliqués.

```
SELECT salesid, qty,
DENSE_RANK() OVER(ORDER BY qty DESC) AS d_rnk,
RANK() OVER(ORDER BY qty DESC) AS rnk
FROM winsales
ORDER BY 2,1;

+---------+-----+-------+-----+
| salesid | qty | d_rnk | rnk |
+---------+-----+-------+-----+
|   10001 |  10 |     5 |   8 |
|   10006 |  10 |     5 |   8 |
|   30001 |  10 |     5 |   8 |
|   40005 |  10 |     5 |   8 |
|   30003 |  15 |     4 |   7 |
|   20001 |  20 |     3 |   4 |
|   20002 |  20 |     3 |   4 |
|   30004 |  20 |     3 |   4 |
```

```
|   10005 |  30 |     2 |   2 |
|   30007 |  30 |     2 |   2 |
|   40001 |  40 |     1 |   1 |
+---------+-----+-------+-----+
```

Notez la différence entre les rangs affectés au même ensemble de lignes lorsque les fonctions DENSE_RANK et RANK sont utilisées côte à côte dans la même requête.

L'exemple suivant partitionne la table en fonction de sellerid, ordonne chaque partition selon la quantité et affecte un rang dense à chaque ligne. Les résultats sont triés une fois que les résultats de la fonction de fenêtrage sont appliqués.

```
SELECT salesid, sellerid, qty,
DENSE_RANK() OVER(PARTITION BY sellerid ORDER BY qty DESC) AS d_rnk
FROM winsales
ORDER BY 2,3,1;

+---------+----------+-----+-------+
| salesid | sellerid | qty | d_rnk |
+---------+----------+-----+-------+
|   10001 |        1 |  10 |     2 |
|   10006 |        1 |  10 |     2 |
|   10005 |        1 |  30 |     1 |
|   20001 |        2 |  20 |     1 |
|   20002 |        2 |  20 |     1 |
|   30001 |        3 |  10 |     4 |
|   30003 |        3 |  15 |     3 |
|   30004 |        3 |  20 |     2 |
|   30007 |        3 |  30 |     1 |
|   40005 |        4 |  10 |     2 |
|   40001 |        4 |  40 |     1 |
+---------+----------+-----+-------+
```

Pour utiliser correctement le dernier exemple, utilisez la commande suivante pour insérer une ligne dans la table WINSALES. Cette ligne possède les mêmes buyerid, sellerid et qtysold qu'une autre ligne. Cela entraîne une égalité entre deux lignes dans le dernier exemple et montre ainsi la différence entre les fonctions DENSE_RANK et RANK.

```
INSERT INTO winsales VALUES(30009, '2/2/2003', 3, 'b', 20, NULL);
```

L'exemple suivant partitionne la table en fonction de buyerid et sellerid, ordonne chaque partition selon la quantité et affecte un rang dense et un rang standard à chaque ligne. Les résultats sont triés après l'application de la fonction de fenêtrage.

```
SELECT salesid, sellerid, qty, buyerid,
DENSE_RANK() OVER(PARTITION BY buyerid, sellerid ORDER BY qty DESC) AS d_rnk,
RANK() OVER (PARTITION BY buyerid, sellerid ORDER BY qty DESC) AS rnk
FROM winsales
ORDER BY rnk;

+---------+----------+-----+---------+-------+-----+
| salesid | sellerid | qty | buyerid | d_rnk | rnk |
+---------+----------+-----+---------+-------+-----+
|   20001 |        2 |  20 | b       |     1 |   1 |
|   30007 |        3 |  30 | c       |     1 |   1 |
|   10006 |        1 |  10 | c       |     1 |   1 |
|   10005 |        1 |  30 | a       |     1 |   1 |
|   20002 |        2 |  20 | c       |     1 |   1 |
|   30009 |        3 |  20 | b       |     1 |   1 |
|   40001 |        4 |  40 | a       |     1 |   1 |
|   30004 |        3 |  20 | b       |     1 |   1 |
|   10001 |        1 |  10 | c       |     1 |   1 |
|   40005 |        4 |  10 | a       |     2 |   2 |
|   30003 |        3 |  15 | b       |     2 |   3 |
|   30001 |        3 |  10 | b       |     3 |   4 |
+---------+----------+-----+---------+-------+-----+
```

Fonction de fenêtrage FIRST_VALUE

Étant donné un ensemble de lignes ordonné, FIRST_VALUE renvoie la valeur de l'expression spécifiée concernant la première ligne du cadre de fenêtrage d'un ensemble de lignes ordonné.

Pour savoir comment sélectionner la dernière ligne du cadre, consultez Fonction de fenêtrage LAST_VALUE.

Syntaxe

```
FIRST_VALUE( expression )[ IGNORE NULLS | RESPECT NULLS ]
OVER (
[ PARTITION BY expr_list ]
[ ORDER BY order_list frame_clause ]
)
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère.

IGNORE NULLS

Lorsque cette option est utilisée avec FIRST_VALUE, la fonction renvoie la première valeur du cadre qui n'est pas NULL (ou NULL si toutes les valeurs sont NULL).

RESPECT NULLS

Indique qu'Amazon Redshift doit contenir des valeurs null pour déterminer la ligne à utiliser. La clause RESPECT NULLS est prise en charge par défaut, si vous ne spécifiez pas IGNORE NULLS.

OVER

Présente les clauses de fenêtrage de la fonction.

PARTITION BY expr_list

Définit la fenêtre de la fonction en termes d'une ou de plusieurs expressions.

ORDER BY order_list

Trie les lignes dans chaque partition. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY trie toute la table. Si vous spécifiez une clause ORDER BY, vous devez également spécifier une frame_clause.

Les résultats de la fonction FIRST_VALUE dépendent de l'ordre des données. Les résultats sont non déterministes dans les cas suivants :

- Quand aucune clause ORDER BY n'est spécifiée et qu'une partition contient deux valeurs différentes pour une expression
- Lorsque l'expression a des valeurs différentes qui correspondent à la même valeur dans la liste ORDER BY.

frame_clause

Si une clause ORDER BY est utilisée pour une fonction d'agrégation, une clause de cadre explicite est requise. La clause de cadre affine l'ensemble de lignes dans la fenêtre d'une fonction, en incluant ou en excluant des ensembles de lignes du résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés. Consultez Récapitulatif de la syntaxe de la fonction de fenêtrage.

Type de retour

Ces fonctions prennent en charge les expressions qui utilisent les types de données primitifs d'Amazon Redshift. Le type de retour est identique au type de données de l'expression.

Exemples

Les exemples suivants utilisent la table VENUE de l'exemple de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

L'exemple suivant renvoie le nombre de places de chaque site dans la table VENUE, avec les résultats classés par capacité (d'élevée à faible). La fonction FIRST_VALUE permet de sélectionner le nom du lieu qui correspond à la première ligne du cadre : dans le cas présent, la ligne comportant le plus grand nombre de places. Les résultats sont partitionnés par État, lorsque la valeur VENUESTATE change, une nouvelle première valeur est donc sélectionnée. Le cadre de fenêtrage est illimité. La même première valeur est donc sélectionnée pour chaque ligne de chaque partition.

Pour la Californie, `Qualcomm Stadium` possède le plus grand nombre de places (`70561`), ce nom est donc la première valeur de toutes les lignes dans la partition `CA`.

```
select venuestate, venueseats, venuename,
first_value(venuename)
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venueseats >0)
order by venuestate;

venuestate | venueseats |           venuename            |         first_value
-----------+------------+--------------------------------
+----------------------------
CA         |      70561 | Qualcomm Stadium               | Qualcomm Stadium
CA         |      69843 | Monster Park                   | Qualcomm Stadium
CA         |      63026 | McAfee Coliseum                | Qualcomm Stadium
CA         |      56000 | Dodger Stadium                 | Qualcomm Stadium
CA         |      45050 | Angel Stadium of Anaheim       | Qualcomm Stadium
CA         |      42445 | PETCO Park                     | Qualcomm Stadium
CA         |      41503 | AT&T Park                      | Qualcomm Stadium
CA         |      22000 | Shoreline Amphitheatre         | Qualcomm Stadium
CO         |      76125 | INVESCO Field                  | INVESCO Field
CO         |      50445 | Coors Field                    | INVESCO Field
DC         |      41888 | Nationals Park                 | Nationals Park
FL         |      74916 | Dolphin Stadium                | Dolphin Stadium
```

```
FL         |      73800 | Jacksonville Municipal Stadium | Dolphin Stadium
FL         |      65647 | Raymond James Stadium          | Dolphin Stadium
FL         |      36048 | Tropicana Field                | Dolphin Stadium
...
```

L'exemple suivant montre l'utilisation de l'option IGNORE NULLS et s'appuie sur l'ajout d'une nouvelle ligne sur la table VENUE :

```
insert into venue values(2000,null,'Stanford','CA',90000);
```

Celle-ci contient une valeur NULL pour la colonne VENUENAME. Répétez à présent la requête FIRST_VALUE indiquée précédemment dans cette section :

```
select venuestate, venueseats, venuename,
first_value(venuename)
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venueseats >0)
order by venuestate;

venuestate | venueseats |         venuename          | first_value
-----------+------------+----------------------------+-------------
CA         |      90000 | NULL                       | NULL
CA         |      70561 | Qualcomm Stadium           | NULL
CA         |      69843 | Monster Park               | NULL
...
```

Du fait que la nouvelle ligne contient la valeur VENUESEATS la plus élevée (90000) et que son VENUENAME a la valeur NULL, la fonction FIRST_VALUE renvoie NULL pour la partition CA. Pour ignorer les lignes comme celle-ci dans l'évaluation de la fonction, ajoutez l'option IGNORE NULLS à l'argument de la fonction :

```
select venuestate, venueseats, venuename,
first_value(venuename) ignore nulls
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venuestate='CA')
order by venuestate;
```

```
venuestate | venueseats |         venuename          |   first_value
-----------+------------+----------------------------+-----------------
CA         |      90000 | NULL                       | Qualcomm Stadium
CA         |      70561 | Qualcomm Stadium           | Qualcomm Stadium
CA         |      69843 | Monster Park               | Qualcomm Stadium
...
```

Fonction de fenêtrage LAG

La fonction de fenêtrage LAG renvoie les valeurs pour une ligne avec un décalage donné au-dessus (avant) de la ligne actuelle dans la partition.

Syntaxe

```
LAG (value_expr [, offset ])
[ IGNORE NULLS | RESPECT NULLS ]
OVER ( [ PARTITION BY window_partition ] ORDER BY window_ordering )
```

Arguments

value_expr

Colonne cible ou expression sur laquelle la fonction opère.

offset

Paramètre facultatif qui spécifie le nombre de lignes avant la ligne actuelle pour lesquelles renvoyer des valeurs. Le décalage peut être un nombre entier constant ou une expression qui a pour valeur un nombre entier. Si vous ne spécifiez pas de décalage, Amazon Redshift utilise 1 comme valeur par défaut. Un décalage de `0` indique la ligne actuelle.

IGNORE NULLS

Spécification facultative qui indique qu'Amazon Redshift doit ignorer les valeurs null pour déterminer les lignes à utiliser. Les valeurs NULL sont incluses si IGNORE NULLS n'est pas répertorié.

> **Note**
>
> Vous pouvez utiliser une expression NVL ou COALESCE pour remplacer les valeurs NULL par une autre valeur. Pour plus d'informations, consultez Fonctions NVL et COALESCE.

RESPECT NULLS

Indique qu'Amazon Redshift doit contenir des valeurs null pour déterminer la ligne à utiliser. La clause RESPECT NULLS est prise en charge par défaut, si vous ne spécifiez pas IGNORE NULLS.

OVER

Spécifie le partitionnement de fenêtrage et d'ordonnancement. La clause OVER ne peut pas contenir de spécification de cadre de fenêtrage.

PARTITION BY window_partition

Argument facultatif qui définit la plage d'enregistrements de chaque groupe de la clause OVER.

ORDER BY window_ordering

Trie les lignes dans chaque partition.

La fonction de fenêtrage LAG prend en charge les expressions qui utilisent l'un des types de données Amazon Redshift. Le type de retour est identique au type value_expr.

Exemples

L'exemple suivant présente la quantité de billets vendus à l'acheteur ayant l'ID d'acheteur 3 et l'heure à laquelle l'acheteur 3 a acheté les billets. Pour comparer chaque vente à la vente précédente de l'acheteur 3, la requête renvoie la quantité précédente vendue pour chaque vente. Dans la mesure où il n'y a aucun achat avant le 16/01/2008, la première quantité précédente vendue a la valeur null :

```
select buyerid, saletime, qtysold,
lag(qtysold,1) over (order by buyerid, saletime) as prev_qtysold
from sales where buyerid = 3 order by buyerid, saletime;

buyerid |      saletime       | qtysold | prev_qtysold
---------+---------------------+---------+--------------
3 | 2008-01-16 01:06:09 |       1 |
3 | 2008-01-28 02:10:01 |       1 |            1
3 | 2008-03-12 10:39:53 |       1 |            1
3 | 2008-03-13 02:56:07 |       1 |            1
3 | 2008-03-29 08:21:39 |       2 |            1
3 | 2008-04-27 02:39:01 |       1 |            2
3 | 2008-08-16 07:04:37 |       2 |            1
3 | 2008-08-22 11:45:26 |       2 |            2
```

```
3 | 2008-09-12 09:11:25 |       1 |            2
3 | 2008-10-01 06:22:37 |       1 |            1
3 | 2008-10-20 01:55:51 |       2 |            1
3 | 2008-10-28 01:30:40 |       1 |            2
(12 rows)
```

Fonction de fenêtrage LAST_VALUE

Pour un ensemble de lignes ordonnées, la fonction LAST_VALUE renvoie la valeur de l'expression par rapport à la dernière ligne du cadre.

Pour savoir comment sélectionner la première ligne du cadre, consultez Fonction de fenêtrage FIRST_VALUE.

Syntaxe

```
LAST_VALUE( expression )[ IGNORE NULLS | RESPECT NULLS ]
OVER (
[ PARTITION BY expr_list ]
[ ORDER BY order_list frame_clause ]
)
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère.

IGNORE NULLS

La fonction renvoie la dernière valeur du cadre qui n'est pas NULL (ou NULL si toutes les valeurs sont NULL).

RESPECT NULLS

Indique qu'Amazon Redshift doit contenir des valeurs null pour déterminer la ligne à utiliser. La clause RESPECT NULLS est prise en charge par défaut, si vous ne spécifiez pas IGNORE NULLS.

OVER

Présente les clauses de fenêtrage de la fonction.

PARTITION BY expr_list

Définit la fenêtre de la fonction en termes d'une ou de plusieurs expressions.

ORDER BY order_list

Trie les lignes dans chaque partition. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY trie toute la table. Si vous spécifiez une clause ORDER BY, vous devez également spécifier une frame_clause.

Les résultats dépendent de l'ordre des données. Les résultats sont non déterministes dans les cas suivants :

- Quand aucune clause ORDER BY n'est spécifiée et qu'une partition contient deux valeurs différentes pour une expression
- Lorsque l'expression a des valeurs différentes qui correspondent à la même valeur dans la liste ORDER BY.

frame_clause

Si une clause ORDER BY est utilisée pour une fonction d'agrégation, une clause de cadre explicite est requise. La clause de cadre affine l'ensemble de lignes dans la fenêtre d'une fonction, en incluant ou en excluant des ensembles de lignes du résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés. Consultez Récapitulatif de la syntaxe de la fonction de fenêtrage.

Type de retour

Ces fonctions prennent en charge les expressions qui utilisent les types de données primitifs d'Amazon Redshift. Le type de retour est identique au type de données de l'expression.

Exemples

Les exemples suivants utilisent la table VENUE de l'exemple de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

L'exemple suivant renvoie le nombre de places de chaque site dans la table VENUE, avec les résultats classés par capacité (d'élevée à faible). La fonction LAST_VALUE permet de sélectionner le nom du lieu qui correspond à la dernière ligne du cadre : dans le cas présent, il s'agit de la ligne présentant le plus petit nombre de places. Les résultats étant partitionnés par État, lorsque la valeur

de VENUESTATE change, une nouvelle dernière valeur est sélectionnée. Comme le cadre de fenêtrage est illimité, la même dernière valeur est sélectionnée pour chaque ligne de chaque partition.

Pour la Californie, `Shoreline Amphitheatre` est renvoyé pour chaque ligne de la partition, car il possède le plus petit nombre de places (`22000`).

```
select venuestate, venueseats, venuename,
last_value(venuename)
over(partition by venuestate
order by venueseats desc
rows between unbounded preceding and unbounded following)
from (select * from venue where venueseats >0)
order by venuestate;

venuestate | venueseats |           venuename            |          last_value
-----------+------------+--------------------------------
+-----------------------------
CA         |      70561 | Qualcomm Stadium               | Shoreline Amphitheatre
CA         |      69843 | Monster Park                   | Shoreline Amphitheatre
CA         |      63026 | McAfee Coliseum                | Shoreline Amphitheatre
CA         |      56000 | Dodger Stadium                 | Shoreline Amphitheatre
CA         |      45050 | Angel Stadium of Anaheim       | Shoreline Amphitheatre
CA         |      42445 | PETCO Park                     | Shoreline Amphitheatre
CA         |      41503 | AT&T Park                      | Shoreline Amphitheatre
CA         |      22000 | Shoreline Amphitheatre         | Shoreline Amphitheatre
CO         |      76125 | INVESCO Field                  | Coors Field
CO         |      50445 | Coors Field                    | Coors Field
DC         |      41888 | Nationals Park                 | Nationals Park
FL         |      74916 | Dolphin Stadium                | Tropicana Field
FL         |      73800 | Jacksonville Municipal Stadium | Tropicana Field
FL         |      65647 | Raymond James Stadium          | Tropicana Field
FL         |      36048 | Tropicana Field                | Tropicana Field
...
```

Fonction de fenêtrage LEAD

La fonction de fenêtrage LEAD renvoie les valeurs pour une ligne avec un décalage donné au-dessous (après) de la ligne actuelle dans la partition.

Syntaxe

```
LEAD (value_expr [, offset ])
```

```
[ IGNORE NULLS | RESPECT NULLS ]
OVER ( [ PARTITION BY window_partition ] ORDER BY window_ordering )
```

Arguments

value_expr

Colonne cible ou expression sur laquelle la fonction opère.

offset

Paramètre facultatif qui spécifie le nombre de lignes sous la ligne actuelle pour lesquelles renvoyer des valeurs. Le décalage peut être un nombre entier constant ou une expression qui a pour valeur un nombre entier. Si vous ne spécifiez pas de décalage, Amazon Redshift utilise 1 comme valeur par défaut. Un décalage de `0` indique la ligne actuelle.

IGNORE NULLS

Spécification facultative qui indique qu'Amazon Redshift doit ignorer les valeurs null pour déterminer les lignes à utiliser. Les valeurs NULL sont incluses si IGNORE NULLS n'est pas répertorié.

> **Note**
>
> Vous pouvez utiliser une expression NVL ou COALESCE pour remplacer les valeurs NULL par une autre valeur. Pour plus d'informations, consultez Fonctions NVL et COALESCE.

RESPECT NULLS

Indique qu'Amazon Redshift doit contenir des valeurs null pour déterminer la ligne à utiliser. La clause RESPECT NULLS est prise en charge par défaut, si vous ne spécifiez pas IGNORE NULLS.

OVER

Spécifie le partitionnement de fenêtrage et d'ordonnancement. La clause OVER ne peut pas contenir de spécification de cadre de fenêtrage.

PARTITION BY window_partition

Argument facultatif qui définit la plage d'enregistrements de chaque groupe de la clause OVER.

ORDER BY window_ordering

Trie les lignes dans chaque partition.

La fonction de fenêtrage LEAD prend en charge les expressions qui utilisent l'un des types de données Amazon Redshift. Le type de retour est identique au type value_expr.

Exemples

L'exemple suivant fournit la commission pour les événements de la table SALES pour les billets ont été vendus sur le 1er janvier 2008 et le 2 janvier 2008 et la commission payée pour la vente des billets de la vente suivante. L'exemple suivant utilise l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

```
SELECT eventid, commission, saletime, LEAD(commission, 1) over ( ORDER BY saletime ) AS
 next_comm
FROM sales
WHERE saletime BETWEEN '2008-01-09 00:00:00' AND '2008-01-10 12:59:59'
LIMIT 10;

+---------+------------+---------------------+-----------+
| eventid | commission |      saletime       | next_comm |
+---------+------------+---------------------+-----------+
|    1664 |       13.2 | 2008-01-09 01:00:21 |      69.6 |
|     184 |       69.6 | 2008-01-09 01:00:36 |     116.1 |
|    6870 |      116.1 | 2008-01-09 01:02:37 |      11.1 |
|    3718 |       11.1 | 2008-01-09 01:05:19 |     205.5 |
|    6772 |      205.5 | 2008-01-09 01:14:04 |      38.4 |
|    3074 |       38.4 | 2008-01-09 01:26:50 |     209.4 |
|    5254 |      209.4 | 2008-01-09 01:29:16 |      26.4 |
|    3724 |       26.4 | 2008-01-09 01:40:09 |      57.6 |
|    5303 |       57.6 | 2008-01-09 01:40:21 |      51.6 |
|    3678 |       51.6 | 2008-01-09 01:42:54 |      43.8 |
+---------+------------+---------------------+-----------+
```

L'exemple suivant indique la différence maximale entre la commission pour les événements figurant dans le tableau des VENTES et la commission payée pour la vente de billets pour la vente ultérieure pour le même événement. Cet exemple montre comment utiliser LEAD avec une clause GROUP BY. Comme les fonctions de fenêtre ne sont pas autorisées dans les clauses d'agrégation, cet exemple utilise une sous-requête. L'exemple suivant utilise l'exemple de base de données TICKIT. Pour de plus amples informations, veuillez consulter Exemple de base de données.

```
SELECT eventid, eventname, max(next_comm_diff) as max_commission_difference
FROM
(
    SELECT sales.eventid, eventname, commission - LEAD(commission, 1) over (ORDER BY
 sales.eventid, saletime) AS next_comm_diff
    FROM sales JOIN event ON sales.eventid = event.eventid
)
GROUP BY eventid, eventname
ORDER BY eventid

LIMIT 10

| eventid | eventname                   | max_commission_difference |
+---------+-----------------------------+---------------------------+
| 1       | Gotterdammerung             | 7.95                      |
| 2       | Boris Godunov               | 227.85                    |
| 3       | Salome                      | 1350.9                    |
| 4       | La Cenerentola (Cinderella) | 790.05                    |
| 5       | Il Trovatore                | 214.05                    |
| 6       | L Elisir d Amore            | 510.9                     |
| 7       | Doctor Atomic               | 180.6                     |
| 9       | The Fly                     | 147                       |
| 10      | Rigoletto                   | 186.6                     |
+---------+-----------------------------+---------------------------+
```

Fonction de fenêtrage LISTAGG

Pour chaque groupe d'une requête, la fonction de fenêtrage LISTAGG trie les lignes du groupe conformément à l'expression ORDER BY, puis concatène les valeurs en une chaîne unique.

Syntaxe

```
LISTAGG( [DISTINCT] expression [, 'delimiter' ] )
[ WITHIN GROUP (ORDER BY order_list) ]
OVER ( [PARTITION BY partition_expression] )
```

Arguments

DISTINCT

(Facultatif) Clause qui supprime toutes les valeurs en double dans l'expression spécifiée avant de procéder à la concaténation. Les espaces de fin étant ignorés, les chaînes `'a'` et `'a '`

sont considérées comme doublons. LISTAGG utilise la première valeur rencontrée. Pour plus d'informations, consultez Signification des blancs de fin.

aggregate_expression

Toute expression valide (par exemple, un nom de colonne) qui fournit les valeurs à regrouper. Les valeurs NULL et les chaînes vides sont ignorées.

delimiter

(Facultatif) Constante de chaîne qui sépare les valeurs concaténées. La valeur par défaut est NULL.

WITHIN GROUP (ORDER BY order_list)

(Facultatif) Clause qui spécifie l'ordre de tri des valeurs regroupées. Déterministe uniquement si ORDER BY fournit un ordonnancement unique. La valeur par défaut consiste à regrouper toutes les lignes et à renvoyer une valeur unique.

OVER

Clause qui spécifie le partitionnement de fenêtrage. La clause OVER ne peut pas contenir d'ordre de fenêtrage ou de spécification de cadre de fenêtrage.

PARTITION BY partition_expression

(Facultatif) Définit la plage d'enregistrements de chaque groupe dans la clause OVER.

Renvoie

VARCHAR(MAX). Si le jeu de résultats est supérieur à la taille de VARCHAR maximale (64 Ko – 1 ou 65535), LISTAGG renvoie l'erreur suivante :

```
Invalid operation: Result size exceeds LISTAGG limit
```

Exemples

Les exemples suivants utilisent la table WINSALES. Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

L'exemple suivant renvoie une liste de vendeurs IDs, classée par numéro de vendeur.

```
select listagg(sellerid)
within group (order by sellerid)
over() from winsales;

  listagg
------------
 11122333344
...
...
 11122333344
 11122333344
   (11 rows)
```

L'exemple suivant renvoie une liste de vendeurs IDs pour l'acheteur B, classée par date.

```
select listagg(sellerid)
within group (order by dateid)
over () as seller
from winsales
where buyerid = 'b' ;

  seller
---------
    3233
    3233
    3233
    3233
```

L’exemple suivant renvoie une liste des dates de ventes de l’acheteur B séparées par des barres virgules.

```
select listagg(dateid,',')
within group (order by sellerid desc,salesid asc)
over () as dates
from winsales
where buyerid  = 'b';

            dates
----------------------------------------
2003-08-02,2004-04-18,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-04-18,2004-02-12
```

```
2003-08-02,2004-04-18,2004-04-18,2004-02-12
```

L'exemple suivant utilise DISTINCT pour renvoyer une liste de dates de vente uniques pour l'acheteur B.

```
select listagg(distinct dateid,',')
within group (order by sellerid desc,salesid asc)
over () as dates
from winsales
where buyerid  = 'b';

           dates
------------------------------
2003-08-02,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-02-12
2003-08-02,2004-04-18,2004-02-12
```

L'exemple suivant renvoie une liste des ventes séparées par des IDs virgules pour chaque numéro d'acheteur.

```
select buyerid,
listagg(salesid,',')
within group (order by salesid)
over (partition by buyerid) as sales_id
from winsales
order by buyerid;

+---------+------------------------+
| buyerid |        sales_id        |
+---------+------------------------+
| a       | 10005,40001,40005      |
| a       | 10005,40001,40005      |
| a       | 10005,40001,40005      |
| b       | 20001,30001,30003,30004 |
| b       | 20001,30001,30003,30004 |
| b       | 20001,30001,30003,30004 |
| b       | 20001,30001,30003,30004 |
| c       | 10001,10006,20002,30007 |
| c       | 10001,10006,20002,30007 |
| c       | 10001,10006,20002,30007 |
| c       | 10001,10006,20002,30007 |
```

```
+---------+-------------------------+
```

Fonction de fenêtrage MAX

La fonction de fenêtrage MAX renvoie le maximum de valeurs d'expression d'entrée. La fonction MAX utilise des valeurs numériques et ignore les valeurs NULL.

Syntaxe

```
MAX ( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list frame_clause ]
)
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère.

ALL

Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression. La valeur par défaut est ALL. DISTINCT n'est pas pris en charge.

OVER

Clause qui spécifie les clauses de fenêtrage des fonctions d'agrégation. La clause OVER différencie les fonctions d'agrégation de fenêtrage des fonctions d'agrégation d'un ensemble normal.

PARTITION BY expr_list

Définit la fenêtre de la fonction MAX en termes d'une ou de plusieurs expressions.

ORDER BY order_list

Trie les lignes dans chaque partition. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY utilise toute la table.

frame_clause

Si une clause ORDER BY est utilisée pour une fonction d'agrégation, une clause de cadre explicite est requise. La clause de cadre affine l'ensemble de lignes dans la fenêtre d'une fonction,

en incluant ou en excluant des ensembles de lignes du résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés. Consultez Récapitulatif de la syntaxe de la fonction de fenêtrage.

Types de données

Accepte n'importe quel type de données comme entrée. Renvoie le même type de données que expression.

Exemples

L'exemple suivant montre l'affichage de l'ID de ventes, la quantité et la quantité maximale dès le début de la fenêtre de données :

```
select salesid, qty,
max(qty) over (order by salesid rows unbounded preceding) as max
from winsales
order by salesid;

salesid | qty | max
---------+-----+-----
10001 |  10 |  10
10005 |  30 |  30
10006 |  10 |  30
20001 |  20 |  30
20002 |  20 |  30
30001 |  10 |  30
30003 |  15 |  30
30004 |  20 |  30
30007 |  30 |  30
40001 |  40 |  40
40005 |  10 |  40
(11 rows)
```

Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

L'exemple suivant montre l'affichage de l'ID de vente, la quantité et la quantité maximale dans un cadre limité :

```
select salesid, qty,
```

```
max(qty) over (order by salesid rows between 2 preceding and 1 preceding) as max
from winsales
order by salesid;

salesid | qty | max
---------+-----+-----
10001 |  10 |
10005 |  30 |  10
10006 |  10 |  30
20001 |  20 |  30
20002 |  20 |  20
30001 |  10 |  20
30003 |  15 |  20
30004 |  20 |  15
30007 |  30 |  20
40001 |  40 |  30
40005 |  10 |  40
(11 rows)
```

Fonction de fenêtrage MEDIAN

Calcule la valeur médiane de la plage de valeurs dans une fenêtre ou une partition. Les valeurs NULL de la plage sont ignorées.

MEDIAN est une fonction de distribution inverse qui suppose un modèle de distribution continue.

Syntaxe

```
MEDIAN ( median_expression )
OVER ( [ PARTITION BY partition_expression ] )
```

Arguments

median_expression

Expression, comme un nom de colonne, qui fournit les valeurs pour lesquelles déterminer la médiane. L'expression doit disposer d'un type de données numériques ou datetime ou être convertible implicitement en une.

OVER

Clause qui spécifie le partitionnement de fenêtrage. La clause OVER ne peut pas contenir d'ordre de fenêtrage ou de spécification de cadre de fenêtrage.

PARTITION BY partition_expression

Facultatif. Expression qui définit la plage d'enregistrements de chaque groupe dans la clause OVER.

Types de données

Le type de retour est déterminé par le type de données de median_expression. Le tableau suivant illustre le type de retour de chaque type de données median_expression.

Type d'entrée	Type de retour
INT2, INT4, INT8, NUMÉRIQUE, DÉCIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATE	DATE

Notes d'utilisation

Si l'argument median_expression est un type de données DECIMAL défini avec la précision maximale de 38 chiffres, il est possible que MEDIAN renvoie un résultat inexact ou une erreur. Si la valeur de retour de la fonction MEDIAN dépasse 38 chiffres, le résultat est tronqué pour s'adapter, ce qui entraîne une perte de précision. Si, au cours de l'interpolation, un résultat intermédiaire dépasse la précision maximale, un dépassement de capacité numérique se produit et la fonction renvoie une erreur. Pour éviter ces conditions, nous vous recommandons d'utiliser un type de données avec une précision inférieure ou l'argument median_expression avec une précision inférieure.

Par exemple, une fonction SUM avec un argument DECIMAL renvoie une précision par défaut de 38 chiffres. L'échelle du résultat est identique à celle de l'argument. Par conséquent, par exemple, une fonction SUM appliquée à une colonne DECIMAL(5,2) renvoie un type de données DECIMAL(38,2).

L'exemple suivant utilise une fonction SUM dans l'argument median_expression d'une fonction MEDIAN. Le type de données de la colonne PRICEPAID est DECIMAL (8,2), la fonction SUM renvoie donc DECIMAL(38,2).

```
select salesid, sum(pricepaid), median(sum(pricepaid))
over() from sales where salesid < 10 group by salesid;
```

Pour éviter une perte potentielle de précision ou une erreur de dépassement de capacité, convertissez le résultat en un type de données DECIMAL avec une précision inférieure, comme dans l'exemple suivant.

```
select salesid, sum(pricepaid), median(sum(pricepaid)::decimal(30,2))
over() from sales where salesid < 10 group by salesid;
```

Exemples

L'exemple suivant calcule le volume de ventes médian de chaque vendeur :

```
select sellerid, qty, median(qty)
over (partition by sellerid)
from winsales
order by sellerid;


sellerid qty median
---------------------------
1  10 10.0
1  10 10.0
1  30 10.0
2  20 20.0
2  20 20.0
3  10 17.5
3  15 17.5
3  20 17.5
3  30 17.5
4  10 25.0
4  40 25.0
```

Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

Fonction de fenêtrage MIN

La fonction de fenêtrage MIN renvoie le minimum de valeurs d'expression d'entrée. La fonction MIN utilise des valeurs numériques et ignore les valeurs NULL.

Syntaxe

```
MIN ( [ ALL ] expression ) OVER
```

```
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list frame_clause ]
)
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère.

ALL

Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression. La valeur par défaut est ALL. DISTINCT n'est pas pris en charge.

OVER

Spécifie les clauses de fenêtrage des fonctions d'agrégation. La clause OVER différencie les fonctions d'agrégation de fenêtrage des fonctions d'agrégation d'un ensemble normal.

PARTITION BY expr_list

Définit la fenêtre de la fonction MIN en termes d'une ou de plusieurs expressions.

ORDER BY order_list

Trie les lignes dans chaque partition. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY utilise toute la table.

frame_clause

Si une clause ORDER BY est utilisée pour une fonction d'agrégation, une clause de cadre explicite est requise. La clause de cadre affine l'ensemble de lignes dans la fenêtre d'une fonction, en incluant ou en excluant des ensembles de lignes du résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés. Consultez Récapitulatif de la syntaxe de la fonction de fenêtrage.

Types de données

Accepte n'importe quel type de données comme entrée. Renvoie le même type de données que expression.

Exemples

L'exemple suivant montre l'affichage de l'ID de ventes, la quantité et la quantité minimale dès le début de la fenêtre de données :

```
select salesid, qty,
min(qty) over
(order by salesid rows unbounded preceding)
from winsales
order by salesid;

salesid | qty | min
---------+-----+-----
10001 |  10 |  10
10005 |  30 |  10
10006 |  10 |  10
20001 |  20 |  10
20002 |  20 |  10
30001 |  10 |  10
30003 |  15 |  10
30004 |  20 |  10
30007 |  30 |  10
40001 |  40 |  10
40005 |  10 |  10
(11 rows)
```

Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

L'exemple suivant montre l'affichage de l'ID de vente, la quantité et la quantité minimale dans un cadre limité :

```
select salesid, qty,
min(qty) over
(order by salesid rows between 2 preceding and 1 preceding) as min
from winsales
order by salesid;

salesid | qty | min
---------+-----+-----
10001 |  10 |
10005 |  30 |  10
```

```
10006 |  10 |  10
20001 |  20 |  10
20002 |  20 |  10
30001 |  10 |  20
30003 |  15 |  10
30004 |  20 |  10
30007 |  30 |  15
40001 |  40 |  20
40005 |  10 |  30
(11 rows)
```

Fonction de fenêtrage NTH_VALUE

La fonction de fenêtrage NTH_VALUE renvoie la valeur d'expression de la ligne spécifiée du cadre de fenêtrage associée à la première ligne de la fenêtre.

Syntaxe

```
NTH_VALUE (expr, offset)
[ IGNORE NULLS | RESPECT NULLS ]
OVER
( [ PARTITION BY window_partition ]
[ ORDER BY window_ordering
                        frame_clause ] )
```

Arguments

expr

Colonne cible ou expression sur laquelle la fonction opère.

offset

Détermine le nombre de lignes associé à la première ligne dans la fenêtre pour laquelle renvoyer l'expression. offset peut être une constante ou une expression et doit être un nombre entier positif qui est supérieur à 0.

IGNORE NULLS

Spécification facultative qui indique qu'Amazon Redshift doit ignorer les valeurs null pour déterminer les lignes à utiliser. Les valeurs NULL sont incluses si IGNORE NULLS n'est pas répertorié.

RESPECT NULLS

Indique qu'Amazon Redshift doit contenir des valeurs null pour déterminer la ligne à utiliser. La clause RESPECT NULLS est prise en charge par défaut, si vous ne spécifiez pas IGNORE NULLS.

OVER

Spécifie le partitionnement, l'ordonnancement et le cadre de fenêtrage.

PARTITION BY window_partition

Définit la plage d'enregistrements de chaque groupe dans la clause OVER.

ORDER BY window_ordering

Trie les lignes dans chaque partition. Si ORDER BY n'est pas spécifié, le cadre par défaut se compose de toutes les lignes de la partition.

frame_clause

Si une clause ORDER BY est utilisée pour une fonction d'agrégation, une clause de cadre explicite est requise. La clause de cadre affine l'ensemble de lignes dans la fenêtre d'une fonction, en incluant ou en excluant des ensembles de lignes du résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés. Consultez Récapitulatif de la syntaxe de la fonction de fenêtrage.

La fonction de fenêtrage NTH_VALUE prend en charge les expressions qui utilisent l'un des types de données Amazon Redshift. Le type de retour est identique au type expr.

Exemples

L'exemple suivant présente le nombre de places dans le troisième plus grand site de Californie, de Floride et de New York, par rapport au nombre de places dans les autres sites de ces États :

```
select venuestate, venuename, venueseats,
nth_value(venueseats, 3)
ignore nulls
over(partition by venuestate order by venueseats desc
rows between unbounded preceding and unbounded following)
as third_most_seats
from (select * from venue where venueseats > 0 and
```

```
venuestate in('CA', 'FL', 'NY'))
order by venuestate;

venuestate |           venuename            | venueseats | third_most_seats
-----------+--------------------------------+------------+------------------
CA         | Qualcomm Stadium               |      70561 |            63026
CA         | Monster Park                   |      69843 |            63026
CA         | McAfee Coliseum                |      63026 |            63026
CA         | Dodger Stadium                 |      56000 |            63026
CA         | Angel Stadium of Anaheim       |      45050 |            63026
CA         | PETCO Park                     |      42445 |            63026
CA         | AT&T Park                      |      41503 |            63026
CA         | Shoreline Amphitheatre         |      22000 |            63026
FL         | Dolphin Stadium                |      74916 |            65647
FL         | Jacksonville Municipal Stadium |      73800 |            65647
FL         | Raymond James Stadium          |      65647 |            65647
FL         | Tropicana Field                |      36048 |            65647
NY         | Ralph Wilson Stadium           |      73967 |            20000
NY         | Yankee Stadium                 |      52325 |            20000
NY         | Madison Square Garden          |      20000 |            20000
(15 rows)
```

Fonction de fenêtrage NTILE

La fonction de fenêtrage NTILE sépare les lignes ordonnées de la partition selon le nombre de groupes de lignes classés de taille égale autant que possible et renvoie le groupe dans lequel se situe une ligne donnée.

Syntaxe

```
NTILE (expr)
OVER (
[ PARTITION BY expression_list ]
[ ORDER BY order_list ]
)
```

Arguments

expr

Nombre de groupes de rang et doit se traduire par une valeur de nombre entier positif (supérieur à 0) pour chaque partition. L'argument expr ne doit pas autoriser la valeur NULL.

OVER

Clause qui spécifie le partitionnement et l'ordonnancement de fenêtrage. La clause OVER ne peut pas contenir de spécification de cadre de fenêtrage.

PARTITION BY window_partition

Facultatif. Plage d'enregistrements de chaque groupe dans la clause OVER.

ORDER BY window_ordering

Facultatif. Expression qui trie les lignes dans chaque partition. Si la clause ORDER BY n'est pas spécifiée, le comportement de rang est identique.

Si ORDER BY ne génère pas d'ordonnancement unique, l'ordre des lignes est non déterministe. Pour plus d'informations, consultez Ordonnancement unique des données pour les fonctions de fenêtrage.

Type de retour

BIGINT

Exemples

L'exemple suivant répartit en quatre groupes de rangs le prix payé pour les billets de Hamlet le 26 août 2008. L'ensemble de résultats est de 17 lignes, classées presque uniformément de 1 à 4 :

```
select eventname, caldate, pricepaid, ntile(4)
over(order by pricepaid desc) from sales, event, date
where sales.eventid=event.eventid and event.dateid=date.dateid and eventname='Hamlet'
and caldate='2008-08-26'
order by 4;

eventname |  caldate   | pricepaid | ntile
----------+------------+-----------+-------
Hamlet    | 2008-08-26 |   1883.00 |     1
Hamlet    | 2008-08-26 |   1065.00 |     1
Hamlet    | 2008-08-26 |    589.00 |     1
Hamlet    | 2008-08-26 |    530.00 |     1
Hamlet    | 2008-08-26 |    472.00 |     1
Hamlet    | 2008-08-26 |    460.00 |     2
Hamlet    | 2008-08-26 |    355.00 |     2
```

```
Hamlet    | 2008-08-26 |    334.00 |     2
Hamlet    | 2008-08-26 |    296.00 |     2
Hamlet    | 2008-08-26 |    230.00 |     3
Hamlet    | 2008-08-26 |    216.00 |     3
Hamlet    | 2008-08-26 |    212.00 |     3
Hamlet    | 2008-08-26 |    106.00 |     3
Hamlet    | 2008-08-26 |    100.00 |     4
Hamlet    | 2008-08-26 |     94.00 |     4
Hamlet    | 2008-08-26 |     53.00 |     4
Hamlet    | 2008-08-26 |     25.00 |     4
(17 rows)
```

Fonction de fenêtrage PERCENT_RANK

Calcule le rang en pourcentage d'une ligne donnée. Le rang en pourcentage est déterminé à l'aide de la formule suivante :

```
(x - 1) / (the number of rows in the window or partition - 1)
```

où x est le rang de la ligne actuelle. Le jeu de données suivant illustre l'utilisation de cette formule :

```
Row# Value Rank Calculation PERCENT_RANK
1 15 1 (1-1)/(7-1) 0.0000
2 20 2 (2-1)/(7-1) 0.1666
3 20 2 (2-1)/(7-1) 0.1666
4 20 2 (2-1)/(7-1) 0.1666
5 30 5 (5-1)/(7-1) 0.6666
6 30 5 (5-1)/(7-1) 0.6666
7 40 7 (7-1)/(7-1) 1.0000
```

La plage de valeur de retour est comprise entre 0 et 1, inclus. La première ligne de n'importe quel jeu dispose d'une fonction PERCENT_RANK spécifiée sur 0.

Syntaxe

```
PERCENT_RANK ()
OVER (
[ PARTITION BY partition_expression ]
[ ORDER BY order_list ]
)
```

Arguments

( )

La fonction ne prend pas d'arguments, mais les parenthèses vides sont obligatoires.

OVER

Clause qui spécifie le partitionnement de fenêtrage. La clause OVER ne peut pas contenir de spécification de cadre de fenêtrage.

PARTITION BY partition_expression

Facultatif. Expression qui définit la plage d'enregistrements de chaque groupe dans la clause OVER.

ORDER BY order_list

Facultatif. Expression permettant de calculer le rang en pourcentage. L'expression doit disposer d'un type de données numériques ou être convertible implicitement en une. Si ORDER BY n'est pas spécifié, la valeur de retour est 0 pour toutes les lignes.

Si ORDER BY ne génère pas d'ordonnancement unique, l'ordre des lignes est non déterministe. Pour plus d'informations, consultez Ordonnancement unique des données pour les fonctions de fenêtrage.

Type de retour

FLOAT8

Exemples

L'exemple suivant calcule le rang en pourcentage des volumes de ventes de chaque vendeur :

```
select sellerid, qty, percent_rank()
over (partition by sellerid order by qty)
from winsales;

sellerid qty  percent_rank
----------------------------------------
1  10.00  0.0
1  10.64  0.5
1  30.37  1.0
3  10.04  0.0
```

```
3  15.15  0.33
3  20.75  0.67
3  30.55  1.0
2  20.09  0.0
2  20.12  1.0
4  10.12  0.0
4  40.23  1.0
```

Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

Fonction de fenêtrage PERCENTILE_CONT

La fonction PERCENTILE_CONT est une fonction de distribution inverse qui suppose un modèle de distribution continue. Elle prend une valeur de centile et une spécification de tri, et renvoie une valeur interpolée qui entre dans la catégorie de la valeur de centile donnée en ce qui concerne la spécification de tri.

PERCENTILE_CONT calcule une interpolation linéaire entre les valeurs après les avoir ordonnées. A l'aide de la valeur de centile `(P)` et le nombre de lignes non null `(N)` dans le groupe d'agrégation, la fonction calcule le nombre de lignes après l'ordonnancement des lignes en fonction de la spécification de tri. Ce nombre de lignes `(RN)` est calculé selon la formule `RN = (1+ (P*(N-1))`. Le résultat de la fonction d'agrégation est calculé par interpolation linéaire entre les valeurs des lignes aux numéros de ligne `CRN = CEILING(RN)` et `FRN = FLOOR(RN)`.

Le résultat final sera le suivant.

Si `(CRN = FRN = RN)` le résultat est `(value of expression from row at RN)`

Sinon, le résultat est le suivant :

`(CRN - RN) * (value of expression for row at FRN) + (RN - FRN) * (value of expression for row at CRN)`.

Vous pouvez uniquement spécifier la clause PARTITION dans la clause OVER. Si la PARTITION est sélectionnée, pour chaque ligne, PERCENTILE_CONT renvoie la valeur qui se situerait dans le centile spécifié parmi un ensemble de valeurs d'une partition donnée.

Syntaxe

```
PERCENTILE_CONT ( percentile )
```

```
WITHIN GROUP (ORDER BY expr)
OVER (  [ PARTITION BY expr_list ]  )
```

Arguments

percentile

Constante numérique comprise entre 0 et 1. Les valeurs NULL sont ignorées dans le calcul.

WITHIN GROUP (ORDER BY expr)

Spécifie date/time les valeurs numériques ou sur lesquelles trier et calculer le percentile.

OVER

Spécifie le partitionnement de fenêtrage. La clause OVER ne peut pas contenir d'ordre de fenêtrage ou de spécification de cadre de fenêtrage.

PARTITION BY expr

Argument facultatif qui définit la plage d'enregistrements de chaque groupe de la clause OVER.

Renvoie

Le type de retour est déterminé par le type de données de l'expression ORDER BY dans la clause WITHIN GROUP. Le tableau suivant illustre le type de retour de chaque type de données d'expression ORDER BY.

Type d'entrée	Type de retour
INT2, INT4, INT8, NUMÉRIQUE, DÉCIMAL	DECIMAL
FLOAT, DOUBLE	DOUBLE
DATE	DATE
TIMESTAMP	TIMESTAMP

Notes d'utilisation

Si l'expression ORDER BY est un type de données DECIMAL défini avec la précision maximale de 38 chiffres, il est possible que PERCENTILE_CONT renvoie un résultat inexact ou une erreur. Si

la valeur de retour de la fonction PERCENTILE_CONT dépasse 38 chiffres, le résultat est tronqué pour s'adapter, ce qui entraîne une perte de précision. Si, au cours de l'interpolation, un résultat intermédiaire dépasse la précision maximale, un dépassement de capacité numérique se produit et la fonction renvoie une erreur. Pour éviter ces conditions, nous vous recommandons d'utiliser un type de données avec une précision inférieure ou l'expression ORDER BY avec une précision inférieure.

Par exemple, une fonction SUM avec un argument DECIMAL renvoie une précision par défaut de 38 chiffres. L'échelle du résultat est identique à celle de l'argument. Par conséquent, par exemple, une fonction SUM appliquée à une colonne DECIMAL(5,2) renvoie un type de données DECIMAL(38,2).

L'exemple suivant utilise une fonction SUM dans la clause ORDER BY d'une fonction PERCENTILE_CONT. Le type de données de la colonne PRICEPAID est DECIMAL (8,2), la fonction SUM renvoie donc DECIMAL(38,2).

```
select salesid, sum(pricepaid), percentile_cont(0.6)
within group (order by sum(pricepaid) desc) over()
from sales where salesid < 10 group by salesid;
```

Pour éviter une perte potentielle de précision ou une erreur de dépassement de capacité, convertissez le résultat en un type de données DECIMAL avec une précision inférieure, comme dans l'exemple suivant.

```
select salesid, sum(pricepaid), percentile_cont(0.6)
within group (order by sum(pricepaid)::decimal(30,2) desc) over()
from sales where salesid < 10 group by salesid;
```

Exemples

Les exemples suivants utilisent la table WINSALES. Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

```
select sellerid, qty, percentile_cont(0.5)
within group (order by qty)
over() as median from winsales;

 sellerid | qty | median
----------+-----+--------
        1 |  10 |   20.0
        1 |  10 |   20.0
```

```
        3 |  10 |   20.0
        4 |  10 |   20.0
        3 |  15 |   20.0
        2 |  20 |   20.0
        3 |  20 |   20.0
        2 |  20 |   20.0
        3 |  30 |   20.0
        1 |  30 |   20.0
        4 |  40 |   20.0
(11 rows)
```

```
select sellerid, qty, percentile_cont(0.5)
within group (order by qty)
over(partition by sellerid) as median from winsales;

 sellerid | qty | median
----------+-----+--------
        2 |  20 |   20.0
        2 |  20 |   20.0
        4 |  10 |   25.0
        4 |  40 |   25.0
        1 |  10 |   10.0
        1 |  10 |   10.0
        1 |  30 |   10.0
        3 |  10 |   17.5
        3 |  15 |   17.5
        3 |  20 |   17.5
        3 |  30 |   17.5
(11 rows)
```

L'exemple suivant applique les fonctions PERCENTILE_CONT et PERCENTILE_DISC sur la vente de billets pour les vendeurs de l'état du Washington.

```
SELECT sellerid, state, sum(qtysold*pricepaid) sales,
percentile_cont(0.6) within group (order by sum(qtysold*pricepaid::decimal(14,2) )
 desc) over(),
percentile_disc(0.6) within group (order by sum(qtysold*pricepaid::decimal(14,2) )
 desc) over()
from sales s, users u
where s.sellerid = u.userid and state = 'WA' and sellerid < 1000
group by sellerid, state;

 sellerid | state |  sales  | percentile_cont | percentile_disc
```

```
----------+-------+---------+-----------------+-----------------
      127 | WA    | 6076.00 |         2044.20 |         1531.00
      787 | WA    | 6035.00 |         2044.20 |         1531.00
      381 | WA    | 5881.00 |         2044.20 |         1531.00
      777 | WA    | 2814.00 |         2044.20 |         1531.00
       33 | WA    | 1531.00 |         2044.20 |         1531.00
      800 | WA    | 1476.00 |         2044.20 |         1531.00
        1 | WA    | 1177.00 |         2044.20 |         1531.00
(7 rows)
```

Fonction de fenêtrage PERCENTILE_DISC

La fonction PERCENTILE_DISC est une fonction de distribution inverse qui suppose un modèle de distribution discrète. Elle prend une valeur de centile et une spécification de tri et renvoie un élément de l'ensemble donné.

Pour une valeur de centile donnée P, PERCENTILE_DISC trie les valeurs de l'expression dans la clause ORDER BY et renvoie la valeur avec la valeur de distribution cumulée la plus petite (concernant la même spécification de tri) supérieure ou égale à P.

Vous pouvez uniquement spécifier la clause PARTITION dans la clause OVER.

Syntaxe

```
PERCENTILE_DISC ( percentile )
WITHIN GROUP (ORDER BY expr)
OVER (  [ PARTITION BY expr_list ]  )
```

Arguments

percentile

Constante numérique comprise entre 0 et 1. Les valeurs NULL sont ignorées dans le calcul.

WITHIN GROUP (ORDER BY expr)

Spécifie date/time les valeurs numériques ou sur lesquelles trier et calculer le percentile.

OVER

Spécifie le partitionnement de fenêtrage. La clause OVER ne peut pas contenir d'ordre de fenêtrage ou de spécification de cadre de fenêtrage.

PARTITION BY expr

Argument facultatif qui définit la plage d'enregistrements de chaque groupe de la clause OVER.

Renvoie

Type de données identique à l'expression ORDER BY dans la clause WITHIN GROUP.

Exemples

Les exemples suivants utilisent la table WINSALES. Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

```
SELECT sellerid, qty, PERCENTILE_DISC(0.5)
WITHIN GROUP (ORDER BY qty)
OVER() AS MEDIAN FROM winsales;

+----------+-----+--------+
| sellerid | qty | median |
+----------+-----+--------+
| 3        | 10  | 20     |
| 1        | 10  | 20     |
| 1        | 10  | 20     |
| 4        | 10  | 20     |
| 3        | 15  | 20     |
| 2        | 20  | 20     |
| 2        | 20  | 20     |
| 3        | 20  | 20     |
| 1        | 30  | 20     |
| 3        | 30  | 20     |
| 4        | 40  | 20     |
+----------+-----+--------+

SELECT sellerid, qty, PERCENTILE_DISC(0.5)
WITHIN GROUP (ORDER BY qty)
OVER(PARTITION BY sellerid) AS MEDIAN FROM winsales;

+----------+-----+--------+
| sellerid | qty | median |
+----------+-----+--------+
| 4        | 10  | 10     |
| 4        | 40  | 10     |
| 3        | 10  | 15     |
```

```
| 3        | 15  | 15     |
| 3        | 20  | 15     |
| 3        | 30  | 15     |
| 2        | 20  | 20     |
| 2        | 20  | 20     |
| 1        | 10  | 10     |
| 1        | 10  | 10     |
| 1        | 30  | 10     |
+----------+-----+--------+
```

Pour rechercher PERCENTILE_DISC(0.25) et PERCENTILE_DISC(0.75) pour la quantité lorsqu'elle est partitionnée par ID de vendeur, utilisez les exemples suivants.

```
SELECT sellerid, qty, PERCENTILE_DISC(0.25)
WITHIN GROUP (ORDER BY qty)
OVER(PARTITION BY sellerid) AS quartile1 FROM winsales;

+----------+-----+-----------+
| sellerid | qty | quartile1 |
+----------+-----+-----------+
| 4        | 10  | 10        |
| 4        | 40  | 10        |
| 2        | 20  | 20        |
| 2        | 20  | 20        |
| 3        | 10  | 10        |
| 3        | 15  | 10        |
| 3        | 20  | 10        |
| 3        | 30  | 10        |
| 1        | 10  | 10        |
| 1        | 10  | 10        |
| 1        | 30  | 10        |
+----------+-----+-----------+

SELECT sellerid, qty, PERCENTILE_DISC(0.75)
WITHIN GROUP (ORDER BY qty)
OVER(PARTITION BY sellerid) AS quartile3 FROM winsales;

+----------+-----+-----------+
| sellerid | qty | quartile3 |
+----------+-----+-----------+
| 3        | 10  | 20        |
| 3        | 15  | 20        |
| 3        | 20  | 20        |
```

```
| 3          | 30  | 20          |
| 4          | 10  | 40          |
| 4          | 40  | 40          |
| 2          | 20  | 20          |
| 2          | 20  | 20          |
| 1          | 10  | 30          |
| 1          | 10  | 30          |
| 1          | 30  | 30          |
+----------+-----+-----------+
```

Fonction de fenêtrage RANK

La fonction de fenêtrage RANK détermine le rang d'une valeur dans un groupe de valeurs, en fonction de l'expression ORDER BY dans la clause OVER. Si la clause PARTITION BY facultative est présente, les rangs sont réinitialisés pour chaque groupe de lignes. Les lignes avec des valeurs égales pour les critères de rang reçoivent le même rang. Amazon Redshift ajoute le nombre de lignes à égalité au rang à égalité pour calculer le rang suivant. Par conséquent, les rangs peuvent ne pas être des numéros consécutifs. Par exemple, si deux lignes sont classées 1, le prochain rang est 3.

La fonction RANK diffère de Fonction de fenêtrage DENSE_RANK sur un point : pour DENSE_RANK, si deux lignes ou plus sont à égalité, il n'y a aucun écart dans la séquence des valeurs classées. Par exemple, si deux lignes sont classées 1, le prochain rang est 2.

Vous pouvez avoir des fonctions de rang avec différentes clauses PARTITION BY et ORDER BY dans la même requête.

Syntaxe

```
RANK () OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list ]
)
```

Arguments

( )

La fonction ne prend pas d'arguments, mais les parenthèses vides sont obligatoires.

OVER

Clauses de fenêtrage de la fonction RANK.

PARTITION BY expr_list

Facultatif. Une ou plusieurs expressions qui définissent le fenêtrage.

ORDER BY order_list

Facultatif. Définit les colonnes sur lesquelles les valeurs de rang sont basées. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY utilise toute la table. Si ORDER BY n'est pas spécifié, la valeur de retour est 1 pour toutes les lignes.

Si ORDER BY ne génère pas d'ordonnancement unique, l'ordre des lignes est non déterministe. Pour plus d'informations, consultez Ordonnancement unique des données pour les fonctions de fenêtrage.

Type de retour

INTEGER

Exemples

L'exemple suivant montre le classement de la table selon la quantité vendue (croissant par défaut) et l'affectation d'un rang à chaque ligne. 1 est la valeur classée la plus élevée. Les résultats sont triés une fois que les résultats de la fonction de fenêtrage sont appliqués:

```
select salesid, qty,
rank() over (order by qty) as rnk
from winsales
order by 2,1;

salesid | qty | rnk
--------+-----+-----
10001 |  10 |  1
10006 |  10 |  1
30001 |  10 |  1
40005 |  10 |  1
30003 |  15 |  5
20001 |  20 |  6
20002 |  20 |  6
30004 |  20 |  6
10005 |  30 |  9
30007 |  30 |  9
40001 |  40 |  11
(11 rows)
```

Notez que la clause ORDER BY externe de cet exemple inclut les colonnes 2 et 1 afin de garantir qu'Amazon Redshift renvoie systématiquement des résultats triés chaque fois que cette requête est exécutée. Par exemple, les lignes avec les ventes IDs 10001 et 10006 ont des valeurs QTY et RNK identiques. L'ordonnancement du résultat final défini par la colonne 1 garantit que la ligne 10001 précède toujours 10006. Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

Dans l'exemple suivant, l'ordonnancement est inversé pour la fonction de fenêtrage (`order by qty desc`). A présent, la valeur de rang la plus élevée s'applique à la valeur QTY la plus élevée.

```
select salesid, qty,
rank() over (order by qty desc) as rank
from winsales
order by 2,1;

 salesid | qty | rank
---------+-----+-----
   10001 |  10 |   8
   10006 |  10 |   8
   30001 |  10 |   8
   40005 |  10 |   8
   30003 |  15 |   7
   20001 |  20 |   4
   20002 |  20 |   4
   30004 |  20 |   4
   10005 |  30 |   2
   30007 |  30 |   2
   40001 |  40 |   1
(11 rows)
```

Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

L'exemple suivant montre le partitionnement de la table en fonction de chaque SELLERID, le classement de chaque partition selon la quantité (par ordre décroissant) et l'affectation d'un rang à chaque ligne. Les résultats sont triés une fois que les résultats de la fonction de fenêtrage sont appliqués.

```
select salesid, sellerid, qty, rank() over
(partition by sellerid
order by qty desc) as rank
```

```
from winsales
order by 2,3,1;

salesid | sellerid | qty | rank
--------+----------+-----+-----
  10001 |        1 |  10 |  2
  10006 |        1 |  10 |  2
  10005 |        1 |  30 |  1
  20001 |        2 |  20 |  1
  20002 |        2 |  20 |  1
  30001 |        3 |  10 |  4
  30003 |        3 |  15 |  3
  30004 |        3 |  20 |  2
  30007 |        3 |  30 |  1
  40005 |        4 |  10 |  2
  40001 |        4 |  40 |  1
(11 rows)
```

Fonction de fenêtrage RATIO_TO_REPORT

Calcule le ratio d'une valeur par rapport à la somme des valeurs dans une fenêtre ou une partition. La valeur de RATIO TO REPORT est déterminée à l'aide de la formule suivante :

`value of` ratio_expression `argument for the current row / sum of` ratio_expression `argument for the window or partition`

Le jeu de données suivant illustre l'utilisation de cette formule :

```
Row# Value Calculation RATIO_TO_REPORT
1 2500 (2500)/(13900) 0.1798
2 2600 (2600)/(13900) 0.1870
3 2800 (2800)/(13900) 0.2014
4 2900 (2900)/(13900) 0.2086
5 3100 (3100)/(13900) 0.2230
```

La plage de valeur de retour est comprise entre 0 et 1, inclus. Si ratio_expression a la valeur NULL, la valeur renvoyée est NULL. Si une valeur dans partition_expression est unique, la fonction retournera 1 pour cette valeur.

Syntaxe

```
RATIO_TO_REPORT ( ratio_expression )
```

```
OVER ( [ PARTITION BY partition_expression ] )
```

Arguments

ratio_expression

Expression, comme un nom de colonne, qui fournit la valeur pour laquelle déterminer le ratio. L'expression doit disposer d'un type de données numériques ou être convertible implicitement en une.

Vous ne pouvez pas utiliser d'autre fonction analytique dans ratio_expression.

OVER

Clause qui spécifie le partitionnement de fenêtrage. La clause OVER ne peut pas contenir d'ordre de fenêtrage ou de spécification de cadre de fenêtrage.

PARTITION BY partition_expression

Facultatif. Expression qui définit la plage d'enregistrements de chaque groupe dans la clause OVER.

Type de retour

FLOAT8

Exemples

Les exemples suivants utilisent la table WINSALES. Pour en savoir plus sur la façon de créer la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

L'exemple suivant calcule la ratio-to-report valeur de chaque ligne de la quantité d'un vendeur par rapport au total de toutes les quantités du vendeur.

```
select sellerid, qty, ratio_to_report(qty)
over()
from winsales
order by sellerid;

sellerid  qty    ratio_to_report
----------------------------------------
1         30     0.13953488372093023
1         10     0.046511627906976744
```

```
1          10     0.046511627906976744
2          20     0.09302325581395349
2          20     0.09302325581395349
3          30     0.13953488372093023
3          20     0.09302325581395349
3          15     0.06976744186046512
3          10     0.046511627906976744
4          10     0.046511627906976744
4          40     0.18604651162790697
```

L'exemple suivant calcule les ratios des volumes de ventes de chaque vendeur, par partition.

```
select sellerid, qty, ratio_to_report(qty)
over(partition by sellerid)
from winsales;

sellerid   qty    ratio_to_report
----------------------------------------
2          20     0.5
2          20     0.5
4          40     0.8
4          10     0.2
1          10     0.2
1          30     0.6
1          10     0.2
3          10     0.13333333333333333
3          15     0.2
3          20     0.26666666666666666
3          30     0.4
```

Fonction de fenêtrage ROW_NUMBER

Attribue le nombre ordinal de la ligne actuelle au sein d'un groupe de lignes, en partant de 1, en fonction de l'expression ORDER BY de la clause OVER. Si la clause PARTITION BY facultative est présente, les nombres ordinaux sont réinitialisés pour chaque groupe de lignes. Les lignes avec des valeurs égales pour les expressions ORDER BY reçoivent des numéros de lignes différentes de manière non déterministe.

Syntaxe

```
ROW_NUMBER() OVER(
  [ PARTITION BY expr_list ]
```

```
  [ ORDER BY order_list ]
)
```

Arguments

( )

La fonction ne prend pas d'arguments, mais les parenthèses vides sont obligatoires.

OVER

Clause de fonction de fenêtrage pour la fonction ROW_NUMBER.

PARTITION BY expr_list

Facultatif. Une ou plusieurs expressions de colonne qui répartissent les résultats dans des ensembles de lignes.

ORDER BY order_list

Facultatif. Une ou plusieurs expressions de colonne qui définissent l'ordre des lignes au sein d'un ensemble. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY utilise toute la table.

Si ORDER BY ne génère pas d'ordonnancement unique ou n'est pas spécifiée, l'ordre des lignes est non déterministe. Pour plus d'informations, consultez Ordonnancement unique des données pour les fonctions de fenêtrage.

Type de retour

BIGINT

Exemples

Les exemples suivants utilisent la table `WINSALES`. Pour obtenir une description de la table `WINSALES`, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

L'exemple suivant ordonne la table par QTY (par ordre croissant), puis attribue un numéro de ligne à chaque ligne. Les résultats sont triés une fois que les résultats de la fonction de fenêtrage sont appliqués.

```
SELECT salesid, sellerid, qty,
ROW_NUMBER() OVER(
   ORDER BY qty ASC) AS row
```

```
FROM winsales
ORDER BY 4,1;

salesid   sellerid   qty   row
---------+----------+-----+-----
   30001 |        3 |  10 |   1
   10001 |        1 |  10 |   2
   10006 |        1 |  10 |   3
   40005 |        4 |  10 |   4
   30003 |        3 |  15 |   5
   20001 |        2 |  20 |   6
   20002 |        2 |  20 |   7
   30004 |        3 |  20 |   8
   10005 |        1 |  30 |   9
   30007 |        3 |  30 |  10
   40001 |        4 |  40 |  11
```

L'exemple suivant présente la partition de la table par SELLERID et classe chaque partition par QTY (en ordre croissant), puis affecte un numéro de ligne à chaque ligne. Les résultats sont triés une fois que les résultats de la fonction de fenêtrage sont appliqués.

```
SELECT salesid, sellerid, qty,
ROW_NUMBER() OVER(
  PARTITION BY sellerid
  ORDER BY qty ASC) AS row_by_seller
FROM winsales
ORDER BY 2,4;

 salesid | sellerid | qty | row_by_seller
---------+----------+-----+-----
   10001 |        1 |  10 |   1
   10006 |        1 |  10 |   2
   10005 |        1 |  30 |   3
   20001 |        2 |  20 |   1
   20002 |        2 |  20 |   2
   30001 |        3 |  10 |   1
   30003 |        3 |  15 |   2
   30004 |        3 |  20 |   3
   30007 |        3 |  30 |   4
   40005 |        4 |  10 |   1
   40001 |        4 |  40 |   2
```

L'exemple suivant montre les résultats lorsque les clauses facultatives ne sont pas utilisées.

```
SELECT salesid, sellerid, qty, ROW_NUMBER() OVER() AS row
FROM winsales
ORDER BY 4,1;

salesid   sellerid   qty   row
---------+----------+-----+-----
   30001 |        3 |  10 |   1
   10001 |        1 |  10 |   2
   10005 |        1 |  30 |   3
   40001 |        4 |  40 |   4
   10006 |        1 |  10 |   5
   20001 |        2 |  20 |   6
   40005 |        4 |  10 |   7
   20002 |        2 |  20 |   8
   30003 |        3 |  15 |   9
   30004 |        3 |  20 |  10
   30007 |        3 |  30 |  11
```

Fonctions de fenêtrage STDDEV_SAMP et STDDEV_POP

Les fonctions de fenêtrage STDDEV_SAMP et STDDEV_POP renvoient l'écart type entre l'échantillon et la population d'un ensemble de valeurs numériques (nombre entier, décimale ou à virgule flottante). Voir aussi Fonctions STDDEV_SAMP et STDDEV_POP.

STDDEV_SAMP et STDDEV sont des synonymes de la même fonction.

Syntaxe

```
STDDEV_SAMP | STDDEV | STDDEV_POP
( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère.

ALL

Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression. La valeur par défaut est ALL. DISTINCT n'est pas pris en charge.

OVER

Spécifie les clauses de fenêtrage des fonctions d'agrégation. La clause OVER différencie les fonctions d'agrégation de fenêtrage des fonctions d'agrégation d'un ensemble normal.

PARTITION BY expr_list

Définit la fenêtre de la fonction en termes d'une ou de plusieurs expressions.

ORDER BY order_list

Trie les lignes dans chaque partition. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY utilise toute la table.

frame_clause

Si une clause ORDER BY est utilisée pour une fonction d'agrégation, une clause de cadre explicite est requise. La clause de cadre affine l'ensemble de lignes dans la fenêtre d'une fonction, en incluant ou en excluant des ensembles de lignes du résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés. Consultez Récapitulatif de la syntaxe de la fonction de fenêtrage.

Types de données

Les types d'argument pris en charge par les fonctions STDDEV sont SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL et DOUBLE PRECISION.

Quel que soit le type de données de l'expression, le type de retour d'une fonction STDDEV est un nombre double précision.

Exemples

L'exemple suivant illustre l'utilisation des fonctions STDDEV_POP et VAR_POP en tant que fonctions de fenêtrage. La requête calcule la variance et l'écart type de la population pour les valeurs PRICEPAID dans la table SALES.

```
select salesid, dateid, pricepaid,
round(stddev_pop(pricepaid) over
(order by dateid, salesid rows unbounded preceding)) as stddevpop,
```

```
round(var_pop(pricepaid) over
(order by dateid, salesid rows unbounded preceding)) as varpop
from sales
order by 2,1;

salesid | dateid | pricepaid | stddevpop | varpop
--------+--------+-----------+-----------+---------
  33095 |   1827 |    234.00 |         0 |       0
  65082 |   1827 |    472.00 |       119 |   14161
  88268 |   1827 |    836.00 |       248 |   61283
  97197 |   1827 |    708.00 |       230 |   53019
 110328 |   1827 |    347.00 |       223 |   49845
 110917 |   1827 |    337.00 |       215 |   46159
 150314 |   1827 |    688.00 |       211 |   44414
 157751 |   1827 |   1730.00 |       447 |  199679
 165890 |   1827 |   4192.00 |      1185 | 1403323
...
```

Les exemples de fonctions d'écart type et de variance peuvent être utilisés de la même manière.

Fonction de fenêtrage SUM

La fonction de fenêtrage SUM renvoie la somme des valeurs de la colonne d'entrée ou de l'expression. La fonction SUM utilise des valeurs numériques et ignore les valeurs NULL.

Syntaxe

```
SUM ( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère.

ALL

Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression. La valeur par défaut est ALL. DISTINCT n'est pas pris en charge.

OVER

Spécifie les clauses de fenêtrage des fonctions d'agrégation. La clause OVER différencie les fonctions d'agrégation de fenêtrage des fonctions d'agrégation d'un ensemble normal.

PARTITION BY expr_list

Définit la fenêtre de la fonction SUM en termes d'une ou de plusieurs expressions.

ORDER BY order_list

Trie les lignes dans chaque partition. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY utilise toute la table.

frame_clause

Si une clause ORDER BY est utilisée pour une fonction d'agrégation, une clause de cadre explicite est requise. La clause de cadre affine l'ensemble de lignes dans la fenêtre d'une fonction, en incluant ou en excluant des ensembles de lignes du résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés. Consultez Récapitulatif de la syntaxe de la fonction de fenêtrage.

Types de données

Les types d'argument pris en charge par la fonction SUM sont SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL et DOUBLE PRECISION.

Les types de retour pris en charge par la fonction SUM sont les suivants :

- Arguments BIGINT for SMALLINT ou INTEGER
- Arguments NUMERIC for BIGINT
- DOUBLE PRECISION pour les arguments à virgule flottante

Exemples

L'exemple suivant montre la création d'une somme cumulée (évolutive) des volumes de ventes classés par date et par ID de ventes :

```
select salesid, dateid, sellerid, qty,
sum(qty) over (order by dateid, salesid rows unbounded preceding) as sum
from winsales
```

```
order by 2,1;

salesid |   dateid   | sellerid | qty | sum
--------+------------+----------+-----+-----
30001 | 2003-08-02 |        3 |  10 |  10
10001 | 2003-12-24 |        1 |  10 |  20
10005 | 2003-12-24 |        1 |  30 |  50
40001 | 2004-01-09 |        4 |  40 |  90
10006 | 2004-01-18 |        1 |  10 | 100
20001 | 2004-02-12 |        2 |  20 | 120
40005 | 2004-02-12 |        4 |  10 | 130
20002 | 2004-02-16 |        2 |  20 | 150
30003 | 2004-04-18 |        3 |  15 | 165
30004 | 2004-04-18 |        3 |  20 | 185
30007 | 2004-09-07 |        3 |  30 | 215
(11 rows)
```

Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

L'exemple suivant montre la création d'une somme cumulée (évolutive) des volumes de ventes par date, le partitionnement des résultats par ID de vendeur et le classement des résultats par date et ID de ventes au sein de la partition :

```
select salesid, dateid, sellerid, qty,
sum(qty) over (partition by sellerid
order by dateid, salesid rows unbounded preceding) as sum
from winsales
order by 2,1;

salesid |   dateid   | sellerid | qty | sum
--------+------------+----------+-----+-----
30001 | 2003-08-02 |        3 |  10 |  10
10001 | 2003-12-24 |        1 |  10 |  10
10005 | 2003-12-24 |        1 |  30 |  40
40001 | 2004-01-09 |        4 |  40 |  40
10006 | 2004-01-18 |        1 |  10 |  50
20001 | 2004-02-12 |        2 |  20 |  20
40005 | 2004-02-12 |        4 |  10 |  50
20002 | 2004-02-16 |        2 |  20 |  40
30003 | 2004-04-18 |        3 |  15 |  25
30004 | 2004-04-18 |        3 |  20 |  45
30007 | 2004-09-07 |        3 |  30 |  75
```

```
(11 rows)
```

L'exemple suivant montre la numérotation séquentielle de toutes les lignes de l'ensemble de résultats, classées en fonction des colonnes SELLERID et SALESID :

```
select salesid, sellerid, qty,
sum(1) over (order by sellerid, salesid rows unbounded preceding) as rownum
from winsales
order by 2,1;

salesid | sellerid |  qty | rownum
--------+----------+------+--------
10001 |        1 |   10 |     1
10005 |        1 |   30 |     2
10006 |        1 |   10 |     3
20001 |        2 |   20 |     4
20002 |        2 |   20 |     5
30001 |        3 |   10 |     6
30003 |        3 |   15 |     7
30004 |        3 |   20 |     8
30007 |        3 |   30 |     9
40001 |        4 |   40 |    10
40005 |        4 |   10 |    11
(11 rows)
```

Pour obtenir une description de la table WINSALES, consultez Exemple de tableau contenant des exemples de fonctions de fenêtrage.

L'exemple suivant montre la numérotation séquentielle de toutes les lignes de l'ensemble de résultats, le partitionnement des résultats par SELLERID et le classement des résultats par SELLERID et SALESID au sein de la partition :

```
select salesid, sellerid, qty,
sum(1) over (partition by sellerid
order by sellerid, salesid rows unbounded preceding) as rownum
from winsales
order by 2,1;

salesid | sellerid | qty | rownum
---------+----------+-----+--------
10001 |        1 |  10 |      1
10005 |        1 |  30 |      2
```

```
10006 |        1 |  10 |      3
20001 |        2 |  20 |      1
20002 |        2 |  20 |      2
30001 |        3 |  10 |      1
30003 |        3 |  15 |      2
30004 |        3 |  20 |      3
30007 |        3 |  30 |      4
40001 |        4 |  40 |      1
40005 |        4 |  10 |      2
(11 rows)
```

Fonctions de fenêtrage VAR_SAMP et VAR_POP

Les fonctions de fenêtrage VAR_SAMP et VAR_POP renvoient la variance entre l'échantillon et la population d'un ensemble de valeurs numériques (nombre entier, décimale ou à virgule flottante). Voir aussi Fonctions VAR_SAMP et VAR_POP.

VAR_SAMP et VARIANCE sont des synonymes de la même fonction.

Syntaxe

```
VAR_SAMP | VARIANCE | VAR_POP
( [ ALL ] expression ) OVER
(
[ PARTITION BY expr_list ]
[ ORDER BY order_list
                        frame_clause ]
)
```

Arguments

expression

Colonne cible ou expression sur laquelle la fonction opère.

ALL

Avec l'argument ALL, la fonction conserve toutes les valeurs en double de l'expression. La valeur par défaut est ALL. DISTINCT n'est pas pris en charge.

OVER

Spécifie les clauses de fenêtrage des fonctions d'agrégation. La clause OVER différencie les fonctions d'agrégation de fenêtrage des fonctions d'agrégation d'un ensemble normal.

PARTITION BY expr_list

Définit la fenêtre de la fonction en termes d'une ou de plusieurs expressions.

ORDER BY order_list

Trie les lignes dans chaque partition. Si aucune clause PARTITION BY n'est spécifiée, ORDER BY utilise toute la table.

frame_clause

Si une clause ORDER BY est utilisée pour une fonction d'agrégation, une clause de cadre explicite est requise. La clause de cadre affine l'ensemble de lignes dans la fenêtre d'une fonction, en incluant ou en excluant des ensembles de lignes du résultat ordonné. La clause de cadre se compose du mot-clé ROWS et des spécificateurs associés. Consultez Récapitulatif de la syntaxe de la fonction de fenêtrage.

Types de données

Les types d'argument pris en charge par les fonctions VARIANCE sont SMALLINT, INTEGER, BIGINT, NUMERIC, DECIMAL, REAL et DOUBLE PRECISION.

Quel que soit le type de données de l'expression, le type de retour d'une fonction VARIANCE est un nombre double précision.

Fonctions d'administration système

Rubriques

- CHANGE_QUERY_PRIORITY
- CHANGE_SESSION_PRIORITY
- CHANGE_USER_PRIORITY
- CURRENT_SETTING
- PG_CANCEL_BACKEND
- PG_TERMINATE_BACKEND
- REBOOT_CLUSTER
- SET_CONFIG

Amazon Redshift prend en charge plusieurs fonctions d'administration système.

CHANGE_QUERY_PRIORITY

CHANGE_QUERY_PRIORITY permet aux super-utilisateurs de modifier la priorité d'une requête en cours d'exécution ou en attente de gestion de la charge de travail (WLM).

Cette fonction permet aux super-utilisateurs de modifier immédiatement la priorité de n'importe quelle requête du système. Seul(e) une requête, un utilisateur ou une séance peut s'exécuter avec la priorité `CRITICAL`.

Syntaxe

```
CHANGE_QUERY_PRIORITY(query_id, priority)
```

Arguments

query_id

L'identificateur de requête de la requête dont la priorité a été modifiée. Nécessite une valeur `INTEGER`.

priority

La nouvelle priorité à attribuer à la requête. Cet argument doit être une chaîne précisant la valeur `CRITICAL`, `HIGHEST`, `HIGH`, `NORMAL`, `LOW` ou `LOWEST`.

Type de retour

Aucun

Exemples

Pour montrer la colonne `query_priority` dans la table système STV_WLM_QUERY_STATE, utilisez l'exemple suivant.

```
SELECT query, service_class, query_priority, state
FROM stv_wlm_query_state WHERE service_class = 101;

+-------+---------------+----------------+---------+
| query | service_class | query_priority |  state  |
+-------+---------------+----------------+---------+
```

```
|  1076 |           101 | Lowest         | Running |
|  1075 |           101 | Lowest         | Running |
+-------+---------------+----------------+---------+
```

Pour montrer les résultats d'un super-utilisateur exécutant la fonction `change_query_priority` pour modifier la priorité et la définir sur `CRITICAL`, utilisez l'exemple suivant.

```
SELECT CHANGE_QUERY_PRIORITY(1076, 'Critical');

+-------------------------------------------------------------------------------+
|                            change_query_priority                              |
+-------------------------------------------------------------------------------+
| Succeeded to change query priority. Priority changed from Lowest to Critical. |
+-------------------------------------------------------------------------------+
```

CHANGE_SESSION_PRIORITY

CHANGE_SESSION_PRIORITY permet aux super-utilisateurs de modifier immédiatement la priorité de n'importe quelle séance du système. Seul(e) une séance, un utilisateur ou une requête peut s'exécuter avec la priorité `CRITICAL`.

Syntaxe

```
CHANGE_SESSION_PRIORITY(pid, priority)
```

Arguments

pid

L'identificateur de processus de la séance dont la priorité a été modifiée. La valeur `-1` fait référence à la séance actuelle. Nécessite une valeur `INTEGER`.

priority

La nouvelle priorité à attribuer à la séance. Cet argument doit être une chaîne précisant la valeur `CRITICAL`, `HIGHEST`, `HIGH`, `NORMAL`, `LOW` ou `LOWEST`.

Type de retour

Aucun

Exemples

Pour renvoyer l'identificateur de processus du processus serveur gérant la session actuelle, utilisez l'exemple suivant.

```
SELECT pg_backend_pid();

+----------------+
| pg_backend_pid |
+----------------+
|          30311 |
+----------------+
```

Dans cet exemple, la priorité est modifiée pour être définie sur `LOWEST` pour la session actuelle.

```
SELECT CHANGE_SESSION_PRIORITY(30311, 'Lowest');

+-------------------------------------------------------------------------------------
+
|                               change_session_priority
 |
+-------------------------------------------------------------------------------------
+
| Succeeded to change session priority. Changed session (pid:30311) priority to lowest.
 |
+-------------------------------------------------------------------------------------
+
```

Dans cet exemple, la priorité est modifiée pour être définie sur `HIGH` pour la session actuelle.

```
SELECT CHANGE_SESSION_PRIORITY(-1, 'High');

+-------------------------------------------------------------------------------------
+
|                                  change_session_priority
           |
+-------------------------------------------------------------------------------------
+
| Succeeded to change session priority. Changed session (pid:30311) priority from
 lowest to high. |
+-------------------------------------------------------------------------------------
+
```

Pour créer une procédure stockée qui modifie une priorité de session, utilisez l'exemple suivant. L'autorisation d'exécuter cette procédure stockée est accordée à l'utilisateur de base de données `test_user`.

```
CREATE OR REPLACE PROCEDURE sp_priority_low(pid IN int, result OUT varchar)
AS $$
BEGIN
  SELECT CHANGE_SESSION_PRIORITY(pid, 'low') into result;
END;
$$ LANGUAGE plpgsql
SECURITY DEFINER;
GRANT EXECUTE ON PROCEDURE sp_priority_low(int) TO test_user;
```

Ensuite, l'utilisateur de base de données nommé `test_user` appelle la procédure.

```
CALL sp_priority_low(pg_backend_pid());

+-----------------------------------------------------+
|                        result                       |
+-----------------------------------------------------+
| Success. Change session (pid:13155) priority to low. |
+-----------------------------------------------------+
```

CHANGE_USER_PRIORITY

CHANGE_SESSION_PRIORITY permet aux super-utilisateurs de modifier la priorité de toutes les requêtes émises par un utilisateur en cours d'exécution ou en attente de gestion de la charge de travail (WLM). Seul(e) un utilisateur, une séance ou une requête peut s'exécuter avec la priorité `CRITICAL`.

Syntaxe

```
CHANGE_USER_PRIORITY(user_name, priority)
```

Arguments

user_name

L'utilisateur de base de données dont la priorité de requête a été modifiée.

priority

La nouvelle priorité à attribuer à toutes les requêtes émises par `user_name`. Cet argument doit être une chaîne avec la valeur `CRITICAL`, `HIGHEST`, `HIGH`, `NORMAL`, `LOW`, `LOWEST` ou `RESET`. Seuls les super-utilisateurs peuvent modifier la priorité et la définir sur `CRITICAL`. La modification de la priorité sur `RESET` supprime le paramètre de priorité pour `user_name`.

Type de retour

Aucun

Exemples

Pour modifier la priorité pour l'utilisateur `analysis_user` sur `LOWEST`, utilisez l'exemple suivant.

```
SELECT CHANGE_USER_PRIORITY('analysis_user', 'lowest');

+-----------------------------------------------------------------------------------+
|                               change_user_priority                                |
+-----------------------------------------------------------------------------------+
| Succeeded to change user priority. Changed user (analysis_user) priority to lowest. |
+-----------------------------------------------------------------------------------+
```

Pour modifier la priorité sur `LOW`, utilisez l'exemple suivant.

```
SELECT CHANGE_USER_PRIORITY('analysis_user', 'low');

+-----------------------------------------------------------------------------------------------------
+
|                                    change_user_priority
        |
+-----------------------------------------------------------------------------------------------------
+
| Succeeded to change user priority. Changed user (analysis_user) priority from Lowest
 to low. |
+-----------------------------------------------------------------------------------------------------
+
```

Pour réinitialiser la priorité, utilisez l'exemple suivant.

```
SELECT CHANGE_USER_PRIORITY('analysis_user', 'reset');
```

```
+-----------------------------------------------------+
|                change_user_priority                 |
+-----------------------------------------------------+
| Succeeded to reset priority for user (analysis_user). |
+-----------------------------------------------------+
```

CURRENT_SETTING

CURRENT_SETTING renvoie la valeur actuelle du paramètre de configuration spécifié.

Cette fonction est équivalente à la commande MONTRER.

Syntaxe

```
current_setting('parameter')
```

L'instruction suivante renvoie la valeur actuelle de la variable de contexte de session spécifiée.

```
current_setting('variable_name')
current_setting('variable_name'[, error_if_undefined])
```

Arguments

paramètre

Valeur de paramètre à afficher. Pour obtenir la liste des paramètres de configuration, consultez Référence de configuration

variable_name

Nom de la variable à afficher. Il doit s'agir d'une constante de type chaîne pour les variables de contexte de session.

error_if_undefined

(Facultatif) Valeur booléenne qui spécifie le comportement si le nom de la variable n'existe pas. Quand error_if_undefined est défini sur TRUE, sa valeur par défaut, Amazon Redshift génère une erreur. Quand error_if_undefined est défini sur FALSE, Amazon Redshift renvoie NULL. Amazon Redshift prend en charge le paramètre error_if_undefined uniquement pour les variables de contexte de session. Cette valeur ne peut pas être utilisée lorsque l'entrée est un paramètre de configuration.

Type de retour

Renvoie une chaîne CHAR ou VARCHAR.

Exemples

Pour renvoyer le paramètre actuel pour le paramètre query_group, utilisez l'exemple suivant.

```
SELECT CURRENT_SETTING('query_group');

+-----------------+
| current_setting |
+-----------------+
| unset           |
+-----------------+
```

Pour renvoyer le paramètre actuel pour la variable app_context.user_id, utilisez l'exemple suivant.

```
SELECT CURRENT_SETTING('app_context.user_id', FALSE);
```

PG_CANCEL_BACKEND

Annule une requête. PG_CANCEL_BACKEND équivaut à la commande ANNULER. Vous pouvez annuler les requêtes exécutées simultanément par votre utilisateur. Les super-utilisateurs peuvent annuler n'importe quelle requête.

Syntaxe

```
pg_cancel_backend( pid )
```

Arguments

pid

ID de processus (PID) de la requête à annuler. Vous ne pouvez pas annuler une requête en spécifiant un ID de requête ; vous devez spécifier l'ID de processus de la requête. Nécessite une valeur INTEGER.

Type de retour

Aucun

Notes d'utilisation

Si les requêtes de plusieurs séances détiennent des verrous sur la même table, vous pouvez utiliser la fonction PG_TERMINATE_BACKEND pour mettre fin à l'une des séances, ce qui oblige toutes les transactions en cours d'exécution dans la séance terminée à libérer tous les verrous et à annuler la transaction. Interrogez la table de catalogue PG__LOCKS afin d'afficher les verrous actuellement détenus. Si vous ne pouvez pas annuler une requête, car elle se trouve dans un bloc de transaction (BEGIN ... END), vous pouvez mettre fin à la séance dans laquelle s'exécute la requête à l'aide de la fonction PG_TERMINATE_BACKEND.

Exemples

Pour annuler une requête en cours d'exécution, récupérez d'abord l'ID de processus de la requête que vous voulez annuler. Pour déterminer le processus IDs de toutes les requêtes en cours d'exécution, exécutez la commande suivante.

```
SELECT pid, TRIM(starttime) AS start,
duration, TRIM(user_name) AS user,
SUBSTRING(query,1,40) AS querytxt
FROM stv_recents
WHERE status = 'Running';

+-----+----------------------+----------+--------+----------------------------+
| pid |       starttime      | duration |  user  |          querytxt          |
+-----+----------------------+----------+--------+----------------------------+
| 802 | 2013-10-14 09:19:03.55 |      132 | dwuser | select venuename from venue |
| 834 | 2013-10-14 08:33:49.47 |  1250414 | dwuser | select * from listing;      |
| 964 | 2013-10-14 08:30:43.29 |   326179 | dwuser | select sellerid from sales  |
+-----+----------------------+----------+--------+----------------------------+
```

Pour annuler la requête dotée de l'ID de processus 802, utilisez l'exemple suivant.

```
SELECT PG_CANCEL_BACKEND(802);
```

PG_TERMINATE_BACKEND

Met fin à une séance. Vous pouvez mettre fin à une séance appartenant à votre utilisateur. Un super-utilisateur peut mettre fin à n'importe quelle séance.

Syntaxe

```
pg_terminate_backend( pid )
```

Arguments

pid

L'ID du processus de la séance pour être arrêté. Nécessite une valeur `INTEGER`.

Type de retour

Aucun

Notes d'utilisation

Si vous êtes sur le point d'atteindre la limite de connexions simultanées, utilisez PG_TERMINATE_BACKEND pour mettre fin aux séances inactives et libérer les connexions. Pour plus d'informations, consultez Limites d'Amazon Redshift.

Si les requêtes de plusieurs séances détiennent des verrous sur la même table, vous pouvez utiliser la fonction PG_TERMINATE_BACKEND pour mettre fin à l'une des séances, ce qui oblige toutes les transactions en cours d'exécution dans la séance terminée à libérer tous les verrous et à annuler la transaction. Interrogez la table de catalogue PG_LOCKS afin d'afficher les verrous actuellement détenus.

Si une requête ne se trouve dans un bloc de transaction (BEGIN... END), vous pouvez annuler la requête à l'aide de la commande ANNULER ou de la fonction PG_CANCEL_BACKEND.

Exemples

Pour interroger la table SVV_TRANSACTIONS afin d'afficher tous les verrous en vigueur pour les transactions actuelles, utilisez l'exemple suivant.

```
SELECT * FROM svv_transactions;

+----------+--------+-------+------+--------------------+-----------------
+----------------------+----------+---------+
| txn_owner | txn_db |  xid  | pid  |      txn_start      |    lock_mode    |
 lockable_object_type | relation | granted |
```

```
+----------+--------+-------+------+--------------------+----------------
+----------------------+----------+---------+
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | AccessShareLock | relation
           |    51940 | true    |
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | AccessShareLock | relation
           |    52000 | true    |
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | AccessShareLock | relation
           |   108623 | true    |
| rsuser    | dev    | 96178 | 8585 | 2017-04-12 20:13:07 | ExclusiveLock   |
 transactionid        |          | true    |
+----------+--------+-------+------+--------------------+----------------
+----------------------+----------+---------+
```

Pour terminer la session détenant les verrous, utilisez l'exemple suivant.

```
SELECT PG_TERMINATE_BACKEND(8585);
```

REBOOT_CLUSTER

Redémarrez le cluster Amazon Redshift sans fermer les connexions au cluster. Vous devez être un super-utilisateur de la base de données pour exécuter cette commande.

Une fois ce redémarrage logiciel terminé, le cluster Amazon Redshift renvoie une erreur à l'application utilisateur et demande à l'application utilisateur de soumettre à nouveau toutes les transactions ou requêtes interrompues par le redémarrage logiciel.

Syntaxe

```
SELECT REBOOT_CLUSTER();
```

SET_CONFIG

Définit un paramètre de configuration sur un nouveau paramètre.

Cette fonction est équivalente à la commande SET dans SQL.

Syntaxe

```
SET_CONFIG('parameter', 'new_value' , is_local)
```

L'instruction suivante définit une variable de contexte de session sur un nouveau paramètre.

```
set_config('variable_name', 'new_value' , is_local)
```

Arguments

paramètre

Paramètre à définir.

variable_name

Nom de la variable à définir.

new_value

Nouvelle valeur du paramètre.

is_local

Si la valeur du paramètre est définie sur true, elle s'applique uniquement à la transaction en cours. Les valeurs valides sont `true` ou `1` et `false` ou `0`.

Type de retour

Renvoie une chaîne CHAR ou VARCHAR.

Exemples

Pour définir la valeur du paramètre `query_group` sur `test` pour la transaction en cours uniquement, utilisez l'exemple suivant.

```
SELECT SET_CONFIG('query_group', 'test', true);

+------------+
| set_config |
+------------+
| test       |
+------------+
```

Pour définir des variables de contexte de session, utilisez l'exemple suivant.

```
SELECT SET_CONFIG('app.username', 'cuddy', FALSE);
```

Fonctions d’informations système

Amazon Redshift prend en charge de nombreuses fonctions d’informations système.

Rubriques

- ACTUEL_ AWS_ACCOUNT
- CURRENT_DATABASE
- CURRENT_NAMESPACE
- CURRENT_SCHEMA
- CURRENT_SCHEMAS
- ARN-SESSION EN COURS
- CURRENT_USER
- CURRENT_USER_ID
- DEFAULT_IAM_ROLE
- GET_MOUNTED_ROLE
- HAS_ASSUMEROLE_PRIVILEGE
- HAS_DATABASE_PRIVILEGE
- HAS_SCHEMA_PRIVILEGE
- HAS_TABLE_PRIVILEGE
- LAST_USER_QUERY_ID
- PG_BACKEND_PID
- PG_GET_COLS
- PG_GET_GRANTEE_BY_IAM_ROLE
- PG_GET_IAM_ROLE_BY_USER
- PG_GET_LATE_BINDING_VIEW_COLS
- PG_GET_SESSION_ROLES
- PG_LAST_COPY_COUNT
- PG_LAST_COPY_ID
- PG_LAST_UNLOAD_ID
- PG_LAST_QUERY_ID

- PG_LAST_UNLOAD_COUNT
- Fonction SLICE_NUM
- USER
- ROLE_IS_MEMBER_OF
- USER_IS_MEMBER_OF
- VERSION

ACTUEL_ AWS_ACCOUNT

Renvoie le AWS compte associé au cluster Amazon Redshift qui a soumis une requête.

Syntaxe

```
current_aws_account
```

Type de retour

Renvoie un entier.

exemple

La requête suivante renvoie le nom de la base de données actuelle.

```
select user, current_aws_account;
current_user | current_account
-------------+---------------
dwuser       | 987654321

(1 row)
```

CURRENT_DATABASE

Renvoie le nom de la base de données à laquelle vous êtes actuellement connecté.

Syntaxe

```
current_database()
```

Type de retour

Renvoie une chaîne CHAR ou VARCHAR.

exemple

La requête suivante renvoie le nom de la base de données actuelle.

```
select current_database();

current_database
------------------
tickit
(1 row)
```

CURRENT_NAMESPACE

Renvoie l'espace de noms du cluster Amazon Redshift actuel. L'espace de noms de cluster Amazon Redshift est l'ID unique du cluster Amazon Redshift.

Syntaxe

```
current_namespace
```

Type de retour

Renvoie une chaîne CHAR ou VARCHAR.

exemple

La requête suivante renvoie le nom de l'espace de noms actuel :

```
select user, current_namespace;
current_user | current_namespace
-------------+--------------------------------------
dwuser       | 86b5169f-01dc-4a6f-9fbb-e2e24359e9a8

(1 row)
```

CURRENT_SCHEMA

Renvoie le nom du schéma devant le chemin de recherche. Ce schéma sera utilisé pour les tables ou autres objets nommés qui sont créés sans spécifier de schéma cible.

Syntaxe

> **Note**
>
> Il s'agit d'une fonction de nœud principal. Cette fonction renvoie une erreur si elle fait référence à une table créée par l'utilisateur, à une table système STL ou STV ou à une vue système SVV ou SVL.

```
current_schema()
```

Type de retour

CURRENT_SCHEMA renvoie une chaîne CHAR ou VARCHAR.

Exemples

La requête suivante renvoie le schéma actuel :

```
select current_schema();

current_schema
---------------
public
(1 row)
```

CURRENT_SCHEMAS

Renvoie un tableau des noms de n'importe quel schéma dans le chemin d'accès actuel. Le chemin de recherche actuelle est défini dans le paramètre search_path.

Syntaxe

> **Note**
>
> Il s'agit d'une fonction de nœud principal. Cette fonction renvoie une erreur si elle fait référence à une table créée par l'utilisateur, à une table système STL ou STV ou à une vue système SVV ou SVL.

```
current_schemas(include_implicit)
```

Argument

include_implicit

Si la valeur est définie sur true, indique que le chemin de recherche doit inclure des schémas de système inclus implicitement. Les valeurs valides sont `true` et `false`. Généralement, s'il est défini sur `true`, ce paramètre renvoie le schéma `pg_catalog` en plus du schéma actuel.

Type de retour

Renvoie une chaîne CHAR ou VARCHAR.

Exemples

L'exemple suivant renvoie les noms des schémas dans le chemin de recherche actuel, sans inclure implicitement les schémas de système :

```
select current_schemas(false);

current_schemas
-----------------
{public}
(1 row)
```

L'exemple suivant renvoie les noms des schémas dans le chemin de recherche actuel, en incluant implicitement les schémas de système :

```
select current_schemas(true);

current_schemas
--------------------
{pg_catalog,public}
(1 row)
```

ARN-SESSION EN COURS

Renvoie l'ARN de l'utilisateur global actuellement autorisé. Les utilisateurs du monde entier se présentent avec la même identité sur les comptes Redshift, les clusters et les groupes de travail sans serveur. Les utilisateurs du monde entier se connectent via le centre d'identité IAM ou via

l'authentification de session basée sur IAM. Les utilisateurs du data lake sont des AWS utilisateurs du monde entier.

Cette fonction est généralement utilisée dans le contexte de l'utilisation de vues multidialectes AWS Glue . Pour plus d'informations sur la gestion des identités avec IAM Identity Center et Redshift, voir Connect Redshift à IAM Identity Center pour offrir aux utilisateurs une expérience d'authentification unique. Pour plus d'informations sur les vues Glue multidialectes, créez des vues dans le catalogue de AWS Glue données.

Syntaxe

```
current_session_arn()
```

Type de retour

Renvoie une chaîne VARCHAR de l'utilisateur authentifié globalement ou une valeur nulle.

Notes d'utilisation

Les utilisateurs locaux ne sont pas pris en charge et entraînent une réponse nulle.

exemple

La requête suivante renvoie le nom de l'ARN de session en cours :

```
SELECT current_session_arn();

current_session_arn
--------------
arn:aws:iam::123456789012:user/user
(1 row)
```

CURRENT_USER

Renvoie le nom de l'utilisateur actuel de la base de données qui s'applique à la vérification des autorisations. Généralement, ce nom d'utilisateur sera identique à celui de l'utilisateur de la séance. Toutefois, cela peut parfois être modifié par les super-utilisateurs.

> **Note**
>
> N'utilisez pas de parenthèses de fin lorsque vous appelez CURRENT_USER.

Syntaxe

```
current_user
```

Type de retour

CURRENT_USER renvoie un type de données NAME et peut être converti en chaîne CHAR ou VARCHAR.

Notes d'utilisation

Si une procédure stockée a été créée à l'aide de l'option SECURITY DEFINER de la commande CREATE_PROCEDURE, lors de l'appel de la fonction CURRENT_USER depuis la procédure stockée, Amazon Redshift renvoie le nom d'utilisateur du propriétaire de la procédure stockée.

exemple

La requête suivante renvoie le nom de l'utilisateur de la base de données actuelle :

```
select current_user;

current_user
--------------
dwuser
(1 row)
```

CURRENT_USER_ID

Renvoie l'identifiant unique de l'utilisateur Amazon Redshift connecté à la séance en cours.

Syntaxe

```
CURRENT_USER_ID
```

Type de retour

La fonction CURRENT_USER_ID renvoie un nombre entier.

Exemples

L'exemple suivant renvoie le nom d'utilisateur et l'ID de l'utilisateur actuel de cette séance :

```
select user, current_user_id;
```

```
 current_user | current_user_id
--------------+-----------------
   dwuser     |               1
(1 row)
```

DEFAULT_IAM_ROLE

Renvoie le rôle IAM par défaut actuellement associé au cluster Amazon Redshift. La fonction ne renvoie rien si aucun rôle IAM par défaut n'est associé.

Syntaxe

```
select default_iam_role();
```

Type de retour

Renvoie une chaîne VARCHAR.

exemple

L'exemple suivant renvoie le rôle IAM par défaut actuellement associé au cluster Amazon Redshift spécifié,

```
select default_iam_role();
              default_iam_role
---------------------------------------------
 arn:aws:iam::123456789012:role/myRedshiftRole
(1 row)
```

GET_MOUNTED_ROLE

Lorsqu'elle est invoquée dans le cadre d'une AWS Glue vue multidialectale, elle permet de renvoyer le rôle IAM utilisé pour monter le schéma ou la base de données de Lake Formation. Le multi-dialecte signifie que le SQL est pris en charge par plusieurs moteurs de requêtes, tels qu'Amazon EMR et Redshift. Pour plus d'informations sur les vues Glue multidialectes, voir Création de vues dans le catalogue de AWS Glue données.

Syntaxe

```
get_mounted_role()
```

Type de retour

Renvoie une chaîne VARCHAR ou une valeur nulle.

Notes d'utilisation

Cette fonction renvoie la valeur null pour tout cas d'utilisation en dehors d'une vue externe de Lake Formation.

exemple

La requête suivante renvoie l'identité pour monter la ressource Lake Formation.

```
CREATE EXTERNAL PROTECTED VIEW external_schema.remote_view AS
SELECT mycol, get_mounted_role() FROM external_schema.remote_table;

mycol | get_mounted_role
---------------------------
1       arn:aws:iam::123456789012:role/salesrole
(1 row)
```

HAS_ASSUMEROLE_PRIVILEGE

Renvoie la valeur booléenne `true` (`t`) si l'utilisateur a le rôle IAM spécifié avec le privilège d'exécuter la commande spécifiée. La fonction renvoie `false` (`f`) si l'utilisateur n'a pas le rôle IAM spécifié avec le privilège d'exécuter la commande spécifiée. Pour plus d'informations sur les privilèges, consultez GRANT.

Syntaxe

```
has_assumerole_privilege( [ user, ] iam_role_arn, cmd_type)
```

Arguments

user

Nom de l'utilisateur pour vérifier les privilèges du rôle IAM. Le comportement par défaut consiste à vérifier l'utilisateur actuel. Les super-utilisateurs et les utilisateurs peuvent utiliser cette fonction. Toutefois, les utilisateurs ne peuvent afficher que leurs propres privilèges.

iam_role_arn

Rôle IAM auquel les privilèges de commande ont été accordés.

cmd_type

Commande pour laquelle l'accès a été accordé. Les valeurs possibles sont les suivantes :

- COPY
- UNLOAD
- EXTERNAL FUNCTION
- CREATE MODEL

Type de retour

BOOLEAN

exemple

La requête suivante confirme que l'utilisateur `reg_user1` a le privilège pour le rôle `Redshift-S3-Read` d'exécuter la commande COPY.

```
select has_assumerole_privilege('reg_user1', 'arn:aws:iam::123456789012:role/Redshift-S3-Read', 'copy');
```

```
has_assumerole_privilege
------------------------
true
(1 row)
```

HAS_DATABASE_PRIVILEGE

Renvoie la valeur `true` si le privilège est spécifié pour l'utilisateur pour la base de données spécifiée. Pour plus d'informations sur les privilèges, consultez GRANT.

Syntaxe

> **Note**
>
> Il s'agit d'une fonction de nœud principal. Cette fonction renvoie une erreur si elle fait référence à une table créée par l'utilisateur, à une table système STL ou STV ou à une vue système SVV ou SVL.

```
has_database_privilege( [ user, ] database, privilege)
```

Arguments

user

Nom de l'utilisateur pour vérifier les privilèges de base de données. Le comportement par défaut consiste à vérifier l'utilisateur actuel.

database

Base de données associée au privilège.

privilege

Privilège à vérifier. Les valeurs possibles sont les suivantes :

- CREATE
- TEMPORARY
- TEMP

Type de retour

Renvoie une chaîne CHAR ou VARCHAR.

exemple

La requête suivante confirme que l'utilisateur GUEST dispose du privilège TEMP sur la base de données TICKIT.

```
select has_database_privilege('guest', 'tickit', 'temp');

has_database_privilege
------------------------
true
(1 row)
```

HAS_SCHEMA_PRIVILEGE

Renvoie la valeur `true` si le privilège est spécifié pour l'utilisateur pour le schéma spécifié. Pour plus d'informations sur les privilèges, consultez GRANT.

Syntaxe

> **Note**
>
> Il s'agit d'une fonction de nœud principal. Cette fonction renvoie une erreur si elle fait référence à une table créée par l'utilisateur, à une table système STL ou STV ou à une vue système SVV ou SVL.

```
has_schema_privilege( [ user, ] schema, privilege)
```

Arguments

user

Nom de l'utilisateur pour vérifier les privilèges de schéma. Le comportement par défaut consiste à vérifier l'utilisateur actuel.

schéma

Schéma associé au privilège.

privilege

Privilège à vérifier. Les valeurs possibles sont les suivantes :

- CREATE
- USAGE
- ALTER
- DROP

Type de retour

Renvoie une chaîne CHAR ou VARCHAR.

exemple

La requête suivante confirme que l'utilisateur GUEST dispose du privilège CREATE concernant le schéma PUBLIC :

```
select has_schema_privilege('guest', 'public', 'create');
```

```
has_schema_privilege
----------------------
true
(1 row)
```

HAS_TABLE_PRIVILEGE

Renvoie `true` si l'utilisateur dispose du privilège spécifié pour la table indiquée et renvoie `false` dans le cas inverse.

Syntaxe

> **Note**
>
> Il s'agit d'une fonction de nœud principal. Cette fonction renvoie une erreur si elle fait référence à une table créée par l'utilisateur, à une table système STL ou STV ou à une vue système SVV ou SVL. Pour plus d'informations sur les privilèges, consultez GRANT.

```
has_table_privilege( [ user, ] table, privilege)
```

Arguments

user

Nom de l'utilisateur pour vérifier les privilèges de table. Le comportement par défaut consiste à vérifier l'utilisateur actuel.

table

Table associée au privilège.

privilege

Privilège à vérifier. Les valeurs possibles sont les suivantes :

- SELECT
- INSERT
- UPDATE
- DELETE

- DROP
- REFERENCES

Type de retour

BOOLEAN

Exemples

La requête suivante trouve que l'utilisateur GUEST ne dispose pas du privilège SELECT sur la table LISTING.

```
select has_table_privilege('guest', 'listing', 'select');

has_table_privilege
---------------------
false
```

La requête suivante répertorie les privilèges des tables, notamment les privilèges de sélection, d'insertion, de mise à jour et de suppression, en utilisant les résultats des tables de catalogue pg_tables et pg_user. Il s'agit uniquement d'un exemple. Il se peut que vous deviez spécifier un nom de schéma et des noms de tables à partir de votre base de données. Pour de plus amples informations, veuillez consulter Interrogation des tables catalogue.

```
SELECT
     tablename
     ,usename
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'select') AS sel
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'insert') AS ins
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'update') AS upd
     ,HAS_TABLE_PRIVILEGE(users.usename, tablename, 'delete') AS del
FROM
(SELECT * from pg_tables
WHERE schemaname = 'public' and tablename in ('event','listing')) as tables
,(SELECT * FROM pg_user) AS users;

tablename | usename   |  sel   |  ins  |  upd  | del
----------+-----------+--------+-------+-------+-------
event     |  john     |  true  | true  | true  | true
event     |  sally    |  false | false | false | false
event     |  elsa     |  false | false | false | false
```

```
listing   |  john     |  true  | true  | true  | true
listing   |  sally    |  false | false | false | false
listing   |  elsa     |  false | false | false | false
```

La requête précédente contient également une jointure croisée. Pour de plus amples informations, veuillez consulter Exemples de clause JOIN. Pour interroger des tables qui ne se trouvent pas dans la schéma `public`, retirez la condition `schemaname` de la clause WHERE et utilisez l'exemple suivant avant d'exécuter votre requête.

```
SET SEARCH_PATH to 'schema_name';
```

LAST_USER_QUERY_ID

Renvoie l'ID de la requête d'utilisateur exécutée le plus récemment au cours de la session en cours. Si aucune requête n'a été exécutée dans la session en cours, last_user_query_id renvoie -1. La fonction ne renvoie pas l'ID de requête pour les requêtes qui s'exécutent exclusivement sur le nœud principal. Pour de plus amples informations, veuillez consulter Fonctions exécutées uniquement sur le nœud principal.

Syntaxe

```
last_user_query_id()
```

Type de retour

Renvoie un entier.

exemple

La requête suivante renvoie l'ID de la dernière requête exécutée par un utilisateur dans la session en cours.

```
select last_user_query_id();
```

Voici les résultats.

```
last_user_query_id
----------------------
    5437
```

```
(1 row)
```

La requête suivante renvoie l'ID de la requête et le texte de la requête exécutée le plus récemment par un utilisateur dans la session en cours.

```
select query_id, query_text from sys_query_history where query_id =
 last_user_query_id();
```

Voici les résultats.

```
 query_id, query_text
---------
+-----------------------------------------------------------------------------------------------
 5556975 | select last_user_query_id() limit 100 --RequestID=<unique request ID>;
 TraceID=<unique trace ID>
```

PG_BACKEND_PID

Renvoie l'ID de processus (PID) du processus serveur gérant la séance actuelle.

> **Note**
>
> Le PID n'est pas unique au monde. Il peut être réutilisé au fil du temps.

Syntaxe

```
pg_backend_pid()
```

Type de retour

Renvoie un entier.

exemple

Vous pouvez faire coïncider PG_BACKEND_PID avec des tables de journal afin de récupérer des informations pour la séance en cours. Par exemple, la requête suivante renvoie l'ID de requête et une partie du texte de la requête pour des requêtes exécutées dans la séance en cours.

```
select query, substring(text,1,40)
```

```
from stl_querytext
where pid =  PG_BACKEND_PID()
order by query desc;

 query |                substring
-------+-----------------------------------------
 14831 | select query, substring(text,1,40) from
 14827 | select query, substring(path,0,80) as pa
 14826 | copy category from 's3://dw-tickit/manif
 14825 | Count rows in target table
 14824 | unload ('select * from category') to 's3
(5 rows)
```

Vous pouvez faire coïncider PG_BACKEND_PID avec la colonne de pid dans les tables de journal suivantes (les exceptions sont entre parenthèses) :

- STL_CONNECTION_LOG
- STL_DDLTEXT
- STL_ERROR
- STL_QUERY
- STL_QUERYTEXT
- STL_SESSIONS (process)
- STL_TR_CONFLICT
- STL_UTILITYTEXT
- STV_ACTIVE_CURSORS
- STV_INFLIGHT
- STV_LOCKS (lock_owner_pid)
- STV_RECENTS (process_id)

PG_GET_COLS

Renvoie les métadonnées de colonne pour une table ou une définition de vue.

Syntaxe

```
pg_get_cols('name')
```

Arguments

nom

Nom de la table ou vue Amazon Redshift. Pour de plus amples informations, veuillez consulter Noms et identificateurs.

Type de retour

VARCHAR

Notes d'utilisation

La fonction PG_GET_COLS renvoie une ligne par colonne dans la table ou la définition de vue. La ligne contient une liste séparée par des virgules contenant le nom du schéma, le nom de la relation, le nom de la colonne, le type de données et le numéro de la colonne. Le formatage du résultat du SQL dépend du client SQL utilisé.

Exemples

Les exemples suivants renvoient les résultats d'une vue nommée `SALES_VW` dans le schéma `public` et d'une table nommée `sales` dans le schéma `mytickit1` créés par l'utilisateur dans la base de données connectée`dev`.

L'exemple suivant renvoie les métadonnées de colonne pour une vue nommée`SALES_VW`.

```
select pg_get_cols('sales_vw');

pg_get_cols
---------------------------------------------------------------
(public,sales_vw,salesid,integer,1)
(public,sales_vw,listid,integer,2)
(public,sales_vw,sellerid,integer,3)
(public,sales_vw,buyerid,integer,4)
(public,sales_vw,eventid,integer,5)
(public,sales_vw,dateid,smallint,6)
(public,sales_vw,qtysold,smallint,7)
(public,sales_vw,pricepaid,"numeric(8,2)",8)
(public,sales_vw,commission,"numeric(8,2)",9)
(public,sales_vw,saletime,"timestamp without time zone",10)
```

L'exemple suivant renvoie les métadonnées de colonne de la `SALES_VW` vue sous forme de tableau.

```
select * from pg_get_cols('sales_vw')
cols(view_schema name, view_name name, col_name name, col_type varchar, col_num int);

view_schema | view_name | col_name   | col_type                    | col_num
------------+-----------+------------+-----------------------------+--------
public      | sales_vw  | salesid    | integer                     |       1
public      | sales_vw  | listid     | integer                     |       2
public      | sales_vw  | sellerid   | integer                     |       3
public      | sales_vw  | buyerid    | integer                     |       4
public      | sales_vw  | eventid    | integer                     |       5
public      | sales_vw  | dateid     | smallint                    |       6
public      | sales_vw  | qtysold    | smallint                    |       7
public      | sales_vw  | pricepaid  | numeric(8,2)                |       8
public      | sales_vw  | commission | numeric(8,2)                |       9
public      | sales_vw  | saletime   | timestamp without time zone |      10
```

L'exemple suivant renvoie les métadonnées de colonne de la SALES table sous forme de schéma `mytickit1` au format de table.

```
select * from pg_get_cols('"mytickit1"."sales"')
cols(view_schema name, view_name name, col_name name, col_type varchar, col_num int);

view_schema | view_name | col_name   | col_type                    | col_num
------------+-----------+------------+-----------------------------+--------
mytickit1   | sales     | salesid    | integer                     |       1
mytickit1   | sales     | listid     | integer                     |       2
mytickit1   | sales     | sellerid   | integer                     |       3
mytickit1   | sales     | buyerid    | integer                     |       4
mytickit1   | sales     | eventid    | integer                     |       5
mytickit1   | sales     | dateid     | smallint                    |       6
mytickit1   | sales     | qtysold    | smallint                    |       7
mytickit1   | sales     | pricepaid  | numeric(8,2)                |       8
mytickit1   | sales     | commission | numeric(8,2)                |       9
mytickit1   | sales     | saletime   | timestamp without time zone |      10
```

PG_GET_GRANTEE_BY_IAM_ROLE

Renvoie tous les utilisateurs et groupes auxquels un rôle IAM spécifié est attribué.

Syntaxe

```
pg_get_grantee_by_iam_role('iam_role_arn')
```

Arguments

iam_role_arn

Rôle IAM pour lequel renvoyer les utilisateurs et les groupes auxquels ce rôle a été attribué.

Type de retour

VARCHAR

Notes d'utilisation

La fonction PG_GET_GRANTEE_BY_IAM_ROLE renvoie une ligne pour chaque utilisateur ou groupe. Chaque ligne contient le nom du bénéficiaire, le type de bénéficiaire et le privilège accordé. Les valeurs possibles pour le type de bénéficiaire sont p pour le public, u pour l'utilisateur et g pour le groupe.

Vous devez être un super-utilisateur pour utiliser cette fonction.

exemple

L'exemple suivant indique que le rôle IAM `Redshift-S3-Write` est accordé à `group1` et `reg_user1`. Les utilisateurs dans `group_1` peuvent spécifier le rôle uniquement pour les opérations COPY, et l'utilisateur `reg_user1` peut spécifier le rôle uniquement pour effectuer des opérations UNLOAD.

```
select pg_get_grantee_by_iam_role('arn:aws:iam::123456789012:role/Redshift-S3-Write');
```

```
  pg_get_grantee_by_iam_role
-----------------------------
 (group_1,g,COPY)
 (reg_user1,u,UNLOAD)
```

L'exemple suivant de la fonction PG_GET_GRANTEE_BY_IAM_ROLE met en forme le résultat sous la forme d'une table.

```
select grantee, grantee_type, cmd_type FROM
 pg_get_grantee_by_iam_role('arn:aws:iam::123456789012:role/Redshift-S3-Write')
 res_grantee(grantee text, grantee_type text, cmd_type text) ORDER BY 1,2,3;
```

```
  grantee  | grantee_type | cmd_type
-----------+--------------+----------
 group_1   | g            | COPY
 reg_user1 | u            | UNLOAD
```

PG_GET_IAM_ROLE_BY_USER

Renvoie tous les rôles IAM et les privilèges de commande accordés à un utilisateur.

Syntaxe

```
pg_get_iam_role_by_user('name')
```

Arguments

nom

Nom de l'utilisateur pour lequel renvoyer les rôles IAM.

Type de retour

VARCHAR

Notes d'utilisation

La fonction PG_GET_IAM_ROLE_BY_USER renvoie une ligne pour chaque ensemble de rôles et de privilèges de commande. La ligne contient une liste séparée par des virgules composée du nom d'utilisateur, du rôle IAM et de la commande.

La valeur `default` dans le résultat indique que l'utilisateur peut spécifier n'importe quel rôle disponible pour exécuter la commande affichée.

Vous devez être un super-utilisateur pour utiliser cette fonction.

exemple

L'exemple suivant indique que l'utilisateur `reg_user1` peut spécifier n'importe quel rôle IAM disponible pour effectuer des opérations COPY. L'utilisateur peut également spécifier le rôle `Redshift-S3-Write` pour les opérations UNLOAD.

```
select pg_get_iam_role_by_user('reg_user1');
```

```
                            pg_get_iam_role_by_user
-------------------------------------------------------------------------------
 (reg_user1,default,COPY)
 (reg_user1,arn:aws:iam::123456789012:role/Redshift-S3-Write,COPY|UNLOAD)
```

L'exemple suivant de la fonction PG_GET_IAM_ROLE_BY_USER met en forme le résultat sous la forme d'une table.

```
select username, iam_role, cmd FROM pg_get_iam_role_by_user('reg_user1')
 res_iam_role(username text, iam_role text, cmd text);
```

```
 username  |                    iam_role                     | cmd
-----------+-------------------------------------------------+------
 reg_user1 | default                                         | None
 reg_user1 | arn:aws:iam::123456789012:role/Redshift-S3-Read | COPY
```

PG_GET_LATE_BINDING_VIEW_COLS

Renvoie les métadonnées de colonne de toutes les vues à liaison tardive de la base de données. Pour plus d'informations, consultez Vues à liaison tardive

Syntaxe

```
pg_get_late_binding_view_cols()
```

Type de retour

VARCHAR

Notes d'utilisation

La fonction `PG_GET_LATE_BINDING_VIEW_COLS` renvoie une ligne pour chaque colonne des vues à liaison tardive. La ligne contient une liste séparée par des virgules contenant le nom du schéma, le nom de la relation, le nom de la colonne, le type de données et le numéro de la colonne.

exemple

L'exemple suivant renvoie les métadonnées de colonne de toutes les vues à liaison tardive.

```
select pg_get_late_binding_view_cols();

pg_get_late_binding_view_cols
------------------------------------------------------------
(public,myevent,eventname,"character varying(200)",1)
(public,sales_lbv,salesid,integer,1)
(public,sales_lbv,listid,integer,2)
(public,sales_lbv,sellerid,integer,3)
(public,sales_lbv,buyerid,integer,4)
(public,sales_lbv,eventid,integer,5)
(public,sales_lbv,dateid,smallint,6)
(public,sales_lbv,qtysold,smallint,7)
(public,sales_lbv,pricepaid,"numeric(8,2)",8)
(public,sales_lbv,commission,"numeric(8,2)",9)
(public,sales_lbv,saletime,"timestamp without time zone",10)
(public,event_lbv,eventid,integer,1)
(public,event_lbv,venueid,smallint,2)
(public,event_lbv,catid,smallint,3)
(public,event_lbv,dateid,smallint,4)
(public,event_lbv,eventname,"character varying(200)",5)
(public,event_lbv,starttime,"timestamp without time zone",6)
```

L'exemple suivant renvoie les métadonnées de colonne de toutes les vues à liaison tardive dans un format de tableau.

```
select * from pg_get_late_binding_view_cols() cols(view_schema name, view_name name,
 col_name name, col_type varchar, col_num int);
view_schema | view_name | col_name   | col_type                    | col_num
------------+-----------+------------+-----------------------------+--------
public      | sales_lbv | salesid    | integer                     |       1
public      | sales_lbv | listid     | integer                     |       2
public      | sales_lbv | sellerid   | integer                     |       3
public      | sales_lbv | buyerid    | integer                     |       4
public      | sales_lbv | eventid    | integer                     |       5
public      | sales_lbv | dateid     | smallint                    |       6
public      | sales_lbv | qtysold    | smallint                    |       7
public      | sales_lbv | pricepaid  | numeric(8,2)                |       8
public      | sales_lbv | commission | numeric(8,2)                |       9
public      | sales_lbv | saletime   | timestamp without time zone |      10
public      | event_lbv | eventid    | integer                     |       1
public      | event_lbv | venueid    | smallint                    |       2
public      | event_lbv | catid      | smallint                    |       3
public      | event_lbv | dateid     | smallint                    |       4
```

```
public      | event_lbv | eventname  | character varying(200)      |       5
public      | event_lbv | starttime  | timestamp without time zone |       6
```

PG_GET_SESSION_ROLES

Renvoie les rôles de session de l'utilisateur actuellement connecté. Les rôles de session d'un utilisateur sont les groupes définis par un fournisseur d'identité (IdP) pour l'utilisateur connecté. Par exemple, un fournisseur d'identité (IdP) tel que Microsoft Azure Active Directory (Azure AD) vérifie l'identité de l'utilisateur et fournit tous les groupes externes auxquels l'utilisateur appartient pendant le processus de connexion de l'utilisateur. Ces groupes externes sont transformés en rôles Amazon Redshift et sont disponibles pendant la session en cours. Ces rôles sont appelés rôles de session. Un administrateur peut accorder des privilèges à un rôle de session de la même façon qu'aux autres rôles Amazon Redshift. Pour plus d'informations sur l'utilisation des rôles, consultez Contrôle d'accès basé sur les rôles (RBAC). Pour obtenir des informations sur la gestion des identités auprès d'un fournisseur d'identité (IdP), consultez Fédération de fournisseurs d'identité natifs pour Amazon Redshift dans le Guide de gestion Amazon Redshift.

Pour consulter les rôles définis dans le catalogue Amazon Redshift, interrogez la vue du système. SVV_ROLES

Syntaxe

```
pg_get_session_roles()
```

Type de retour

Ensemble de lignes composé de deux valeurs. La première valeur comporte deux parties séparées par deux points (:) et contient `idp-namespace:role-name`. `idp-namespace` est l'espace de noms du fournisseur d'identité (IdP). `role-name` est le nom du groupe externe figurant dans le fournisseur d'identité (IdP). La deuxième valeur contient un `role-id` qui est l'identifiant du rôle.

Notes d'utilisation

La fonction `PG_GET_SESSION_ROLES` renvoie une ligne pour chaque rôle de session renvoyé.

Exemples

L'exemple suivant renvoie une ligne pour chaque rôle à partir du fournisseur d'identité Azure Active Directory. Les colonnes renvoyées sont converties en `sess_roles` avec des colonnes `name` et

`roleid`. Chaque `name` se compose de l'espace de noms Azure Active Directory et d'un nom de groupe dans Azure Active Directory.

```
SELECT * FROM pg_get_session_roles() AS sess_roles(name name, roleid integer);

name                   roleid
--------------------------------
my_aad:test_group_1    106204
my_aad:test_group_2    106205
my_aad:test_group_3    106206
my_aad:test_group_4    106207
my_aad:test_group_5    106208
```

L'exemple suivant renvoie une ligne pour chaque groupe IAM dont l'utilisateur IAM actuellement connecté est membre. Les colonnes renvoyées sont converties en `sess_roles` avec des colonnes `name` et `roleid`. Chaque `name` se compose de l'espace de noms IAM et du nom du groupe IAM.

```
SELECT * FROM pg_get_session_roles() AS sess_roles(name name, roleid integer);

name                   roleid
--------------------------------
IAM:myGroup            110332
```

PG_LAST_COPY_COUNT

Renvoie le nombre de lignes qui ont été chargées par la dernière commande COPY exécutée au cours de la séance actuelle. PG_LAST_COPY_COUNT est mis à jour avec le dernier ID de COPY, qui est l'ID de la requête de la commande COPY ayant commencé le processus de chargement, même en cas d'échec de la charge. L'ID de requête et l'ID de COPY sont mis à jour lorsque la commande COPY commence le processus de chargement.

Si la commande COPY échoue en raison d'une erreur de syntaxe ou de privilèges insuffisants, l'ID de COPY n'est pas mis à jour et PG_LAST_COPY_COUNT renvoie le nombre de la commande COPY précédente. Si aucune commande COPY n'a été exécutée dans la séance en cours, ou en cas d'échec de la dernière commande COPY pendant le chargement, PG_LAST_COPY_COUNT renvoie 0. Pour plus d'informations, consultez PG_LAST_COPY_ID.

Syntaxe

```
pg_last_copy_count()
```

Type de retour

Renvoie BIGINT.

exemple

La requête suivante renvoie le nombre de lignes chargées par la dernière commande COPY dans la séance en cours.

```
select pg_last_copy_count();

pg_last_copy_count
--------------------
             192497
(1 row)
```

PG_LAST_COPY_ID

Renvoie l'ID de requête de la commande COPY exécutée le plus récemment au cours de la séance actuelle. Si aucune commande COPY n'a été exécutée dans la séance en cours, PG_LAST_COPY_ID renvoie -1.

La valeur de PG_LAST_COPY_ID est mise à jour lorsque la commande COPY commence le processus de chargement. Si la commande COPY échoue en raison de données de chargement non valides, l'ID de COPY est mise à jour, vous pouvez donc utiliser PG_LAST_COPY_ID lorsque vous interrogez la table STL_LOAD_ERRORS. Si la transaction COPY est annulée, l'ID de COPY n'est pas mis à jour.

L'ID de COPY n'est pas mis à jour si la commande COPY échoue en raison d'une erreur qui se produit avant que le processus de chargement commence, par exemple une erreur de syntaxe, une erreur d'accès, des informations d'identification non valides ou des privilèges insuffisants. L'ID de COPY n'est pas mis à jour en cas d'échec de COPY au cours de l'étape de compression d'analyse, qui commence après une connexion réussie, mais avant la charge des données.

Syntaxe

```
pg_last_copy_id()
```

Type de retour

Renvoie un entier.

exemple

La requête suivante renvoie l'ID de requête de la dernière commande COPY dans la séance en cours.

```
select pg_last_copy_id();

pg_last_copy_id
---------------
          5437
(1 row)
```

La requête suivante joint STL_LOAD_ERRORS à STL_LOADERROR_DETAIL pour afficher les erreurs détaillées qui se sont produites pendant la charge la plus récente de la séance en cours :

```
select d.query, substring(d.filename,14,20),
d.line_number as line,
substring(d.value,1,16) as value,
substring(le.err_reason,1,48) as err_reason
from stl_loaderror_detail d, stl_load_errors le
where d.query = le.query
and d.query = pg_last_copy_id();

 query |    substring      | line |  value   |                    err_reason
-------+-------------------+------+----------
+-----------------------------------------------
    558| allusers_pipe.txt |  251 | 251      | String contains invalid or unsupported
 UTF8 code
    558| allusers_pipe.txt |  251 | ZRU29FGR | String contains invalid or unsupported
 UTF8 code
    558| allusers_pipe.txt |  251 | Kaitlin  | String contains invalid or unsupported
 UTF8 code
    558| allusers_pipe.txt |  251 | Walter   | String contains invalid or unsupported
 UTF8 code
```

PG_LAST_UNLOAD_ID

Renvoie l'ID de requête de la commande UNLOAD exécutée le plus récemment dans la séance en cours. Si aucune commande UNLOAD n'a été exécutée dans la séance en cours, PG_LAST_UNLOAD_ID renvoie -1.

La valeur de PG_LAST_UNLOAD_ID est mise à jour lorsque la commande UNLOAD commence le processus de chargement. Si la commande UNLOAD échoue en raison de données de chargement non valides, l'ID UNLOAD est mis à jour de sorte que vous pouvez l'utiliser pour des recherches ultérieures. Si la transaction UNLOAD est annulée, l'ID UNLOAD n'est pas mis à jour.

L'ID UNLOAD n'est pas mis à jour si la commande UNLOAD échoue en raison d'une erreur qui se produit avant que le processus de chargement commence, par exemple une erreur de syntaxe, une erreur d'accès, des informations d'identification non valides ou des privilèges insuffisants.

Syntaxe

```
PG_LAST_UNLOAD_ID()
```

Type de retour

Renvoie un entier.

exemple

La requête suivante renvoie l'ID de requête de la dernière commande UNLOAD dans la séance en cours.

```
select PG_LAST_UNLOAD_ID();

PG_LAST_UNLOAD_ID
---------------
          5437
(1 row)
```

PG_LAST_QUERY_ID

Renvoie l'ID de la requête exécutée le plus récemment au cours de la séance actuelle. Si aucune requête n'a été exécutée dans la séance en cours, PG_LAST_QUERY_ID renvoie -1. PG_LAST_QUERY_ID ne renvoie pas l'ID de requête pour les requêtes qui s'exécutent exclusivement sur le nœud principal. Pour plus d'informations, consultez Fonctions exécutées uniquement sur le nœud principal.

Syntaxe

```
pg_last_query_id()
```

Type de retour

Renvoie un entier.

exemple

La requête suivante renvoie l'ID de la dernière requête exécutée dans la séance en cours.

```
select pg_last_query_id();
```

Voici les résultats.

```
pg_last_query_id
----------------
           5437
(1 row)
```

La requête suivante renvoie l'ID de la requête et le texte de la requête exécutée le plus récemment dans la séance en cours.

```
select query, trim(querytxt) as sqlquery
from stl_query
where query = pg_last_query_id();
```

Voici les résultats.

```
query | sqlquery
------+-------------------------------------------------
 5437 | select name, loadtime from stl_file_scan where loadtime > 1000000;
(1 rows)
```

PG_LAST_UNLOAD_COUNT

Renvoie le nombre de lignes qui ont été déchargées par la dernière commande UNLOAD exécutée au cours de la séance actuelle. PG_LAST_UNLOAD_COUNT est mis à jour avec l'ID de requête de la dernière commande UNLOAD, même si l'opération a échoué. L'ID de requête est mis à jour lorsque la commande UNLOAD est exécutée. Si UNLOAD échoue en raison d'une erreur de syntaxe ou de privilèges insuffisants, PG_LAST_UNLOAD_COUNT renvoie le nombre de la commande UNLOAD précédente. Si aucune commande UNLOAD na été exécutée dans la séance en cours, ou en cas d'échec de la dernière commande UNLOAD pendant l'opération de déchargement, PG_LAST_UNLOAD_COUNT renvoie 0.

Syntaxe

```
pg_last_unload_count()
```

Type de retour

Renvoie BIGINT.

exemple

La requête suivante renvoie le nombre de lignes déchargées par la dernière commande UNLOAD dans la séance en cours.

```
select pg_last_unload_count();

pg_last_unload_count
--------------------
             192497
(1 row)
```

Fonction SLICE_NUM

Renvoie un nombre entier correspondant au nombre de tranches dans le cluster où se trouvent les données d'une ligne. SLICE_NUM ne prend aucun paramètre.

Syntaxe

```
SLICE_NUM()
```

Type de retour

La fonction SLICE_NUM renvoie un nombre entier.

Exemples

L'exemple suivant illustre les tranches contenant des données pour les dix premières lignes EVENT de la table EVENTS :

```
select distinct eventid, slice_num() from event order by eventid limit 10;

 eventid | slice_num
---------+-----------
```

```
       1 |         1
       2 |         2
       3 |         3
       4 |         0
       5 |         1
       6 |         2
       7 |         3
       8 |         0
       9 |         1
      10 |         2
(10 rows)
```

L'exemple suivant renvoie un code (10000) pour illustrer qu'une requête sans instruction FROM s'exécute sur le nœud principal :

```
select slice_num();
slice_num
-----------
10000
(1 row)
```

USER

Synonyme de CURRENT_USER. Consultez CURRENT_USER.

ROLE_IS_MEMBER_OF

Renvoie la valeur true si le rôle est membre d'un autre rôle. Les super-utilisateurs peuvent vérifier l'appartenance de tous les rôles. Les utilisateurs standard disposant de l'autorisation ACCESS SYSTEM TABLE peuvent vérifier l'appartenance de tous les utilisateurs. Dans le cas contraire, les utilisateurs standard ne peuvent vérifier que les rôles auxquels ils ont accès. Amazon Redshift provoque une erreur si les rôles fournis n'existent pas ou si l'utilisateur actuel n'a pas accès au rôle.

Syntaxe

```
role_is_member_of( role_name,  granted_role_name)
```

Arguments

role_name

Nom du rôle.

granted_role_name

Nom du rôle accordé.

Type de retour

Renvoie une valeur BOOLÉENNE.

exemple

La requête suivante confirme que le rôle n'est pas membre de role1 ou de role2.

```
SELECT role_is_member_of('role1', 'role2');

 role_is_member_of
-------------------
             False
```

USER_IS_MEMBER_OF

Renvoie la valeur true si l'utilisateur est membre d'un groupe ou rôle. Les super-utilisateurs peuvent vérifier l'appartenance de tous les utilisateurs. Les utilisateurs standard membres du rôle sys:secadmin ou sys:superuser peuvent vérifier l'appartenance de tous les utilisateurs. Dans le cas contraire, les utilisateurs standard peuvent uniquement se vérifier eux-mêmes. Amazon Redshift envoie une erreur si les identités fournies n'existent pas ou si l'utilisateur actuel n'a pas accès au rôle.

Syntaxe

```
user_is_member_of( user_name,  role_name | group_name)
```

Arguments

user_name

Le nom de l'utilisateur.

role_name

Nom du rôle.

nom_groupe

Nom du groupe.

Type de retour

Renvoie une valeur BOOLÉENNE.

exemple

La requête suivante confirme que l'utilisateur n'est pas membre de role1.

```
SELECT user_is_member_of('reguser', 'role1');

 user_is_member_of
-------------------
           False
```

VERSION

La fonction VERSION renvoie les détails relatifs à la version actuellement installée, avec des informations de version Amazon Redshift spécifiques à la fin.

> **Note**
>
> Il s'agit d'une fonction de nœud principal. Cette fonction renvoie une erreur si elle fait référence à une table créée par l'utilisateur, à une table système STL ou STV ou à une vue système SVV ou SVL.

Syntaxe

```
VERSION()
```

Type de retour

Renvoie une chaîne CHAR ou VARCHAR.

Exemples

L'exemple suivant montre les informations de version du cluster actif :

```
select version();
```

```
 version
```

```
----------------------------------------------------------------------------------------------

PostgreSQL 8.0.2 on i686-pc-linux-gnu, compiled by GCC gcc (GCC) 3.4.2 20041017 (Red
Hat 3.4.2-6.fc3), Redshift 1.0.12103
```

Où `1.0.12103` est le numéro de version du cluster.

> **Note**
>
> Pour forcer la mise à jour de votre cluster vers la dernière version disponible, modifiez votre fenêtre de maintenance.

Mots réservés

La liste suivante recense les mots réservés Amazon Redshift. Vous pouvez utiliser les mots réservés avec des identificateurs délimités (guillemets doubles).

> **Note**
>
> Bien que START et CONNECT ne soient pas des mots réservés, utilisez des identificateurs délimités ou AS si vous utilisez START et CONNECT comme alias de table dans votre requête afin d'éviter tout échec à l'exécution.

Pour de plus amples informations, veuillez consulter Noms et identificateurs.

```
AES128
AES256
ALL
ALLOWOVERWRITE
ANALYSE
ANALYZE
AND
ANY
ARRAY
AS
ASC
AUTHORIZATION
AZ64
BACKUP
```

```
BETWEEN
BINARY
BLANKSASNULL
BOTH
BYTEDICT
BZIP2
CASE
CAST
CHECK
COLLATE
COLUMN
CONSTRAINT
CREATE
CREDENTIALS
CROSS
CURRENT_DATE
CURRENT_TIME
CURRENT_TIMESTAMP
CURRENT_USER
CURRENT_USER_ID
DEFAULT
DEFERRABLE
DEFLATE
DEFRAG
DELTA
DELTA32K
DESC
DISABLE
DISTINCT
DO
ELSE
EMPTYASNULL
ENABLE
ENCODE
ENCRYPT
ENCRYPTION
END
EXCEPT
EXPLICIT
FALSE
FOR
FOREIGN
FREEZE
FROM
```

```
FULL
GLOBALDICT256
GLOBALDICT64K
GRANT
GROUP
GZIP
HAVING
IDENTITY
IGNORE
ILIKE
IN
INITIALLY
INNER
INTERSECT
INTERVAL
INTO
IS
ISNULL
JOIN
LEADING
LEFT
LIKE
LIMIT
LOCALTIME
LOCALTIMESTAMP
LUN
LUNS
LZO
LZOP
MINUS
MOSTLY16
MOSTLY32
MOSTLY8
NATURAL
NEW
NOT
NOTNULL
NULL
NULLS
OFF
OFFLINE
OFFSET
OID
OLD
```

```
ON
ONLY
OPEN
OR
ORDER
OUTER
OVERLAPS
PARALLEL
PARTITION
PERCENT
PERMISSIONS
PIVOT
PLACING
PRIMARY
RAW
READRATIO
RECOVER
REFERENCES
REJECTLOG
RESORT
RESPECT
RESTORE
RIGHT
SELECT
SESSION_USER
SIMILAR
SNAPSHOT
SOME
SYSDATE
SYSTEM
TABLE
TAG
TDES
TEXT255
TEXT32K
THEN
TIMESTAMP
TO
TOP
TRAILING
TRUE
TRUNCATECOLUMNS
UNION
UNIQUE
```

```
UNNEST
UNPIVOT
USER
USING
VERBOSE
WALLET
WHEN
WHERE
WITH
WITHOUT
```

Informations de référence sur les tables et les vues système

Amazon Redshift comporte de nombreuses tables et vues système contenant des informations sur le fonctionnement du système. Vous pouvez interroger ces tables et vues système de la même manière que vous interrogez toute autre table de base de données. Cette section présente des exemples de requêtes de table système et explique :

- comment les différents types de tables et de vues système sont générés ;
- quels types d'informations vous pouvez obtenir à partir de ces tables ;
- comment joindre des tables système Amazon Redshift à des tables de catalogue ;
- comment gérer la croissance des fichiers journaux de table système.

Certaines tables système ne peuvent être utilisées que par le AWS personnel à des fins de diagnostic. Les sections suivantes décrivent les tables système qui peuvent être interrogées par les administrateurs système ou d'autres utilisateurs de base de données afin d'obtenir des informations utiles.

> **Note**
>
> Les tables système ne sont pas incluses dans les sauvegardes de cluster automatiques ou manuelles (instantanés). Les vues du système STL conservent sept jours d'historique des journaux. La conservation des journaux ne nécessite aucune intervention du client, mais si vous souhaitez conserver les données du journal pendant plus de 7 jours, vous devrez les copier régulièrement sur d'autres tables ou les décharger sur Amazon S3.

Rubriques

- Types de tables et de vues système
- Visibilité des données dans les tables et vues système
- Migration de requêtes provisionnées uniquement vers des requêtes SYS Monitoring View
- Amélioration du suivi des identificateurs de requêtes à l'aide des vues de surveillance SYS
- Identifiants de requête, de processus et de session dans la table système
- Vues de métadonnées SVV

- Vues de surveillance SYS
- Cartographie des vues système pour la migration vers les vues de surveillance SYS
- Surveillance système (clusters mis en service uniquement)
- Tables catalogue système

Types de tables et de vues système

Il existe plusieurs types de tables et de vues système :

- Les vues SVV contiennent des informations sur les objets de base de données faisant référence à des tables STV temporaires.
- Les vues SYS permettent de surveiller l'utilisation de requêtes et de charges de travail pour les clusters mis en service et les groupes de travail sans serveur.
- Les vues STL sont générées à partir des journaux stockés durablement sur le disque pour fournir un historique du système.
- Les tables STV sont des tables système virtuelles contenant les instantanés des données système actuelles. Elles sont basées sur des données en mémoire temporaires et ne sont pas stockées durablement sous forme de journaux sur disque ou de tables standard.
- Les vues SVCS fournissent des détails sur les requêtes effectuées sur les clusters principaux et les clusters de mise à l'échelle de simultanéité.
- Les détails fournis par les vues SVL portent sur les requêtes exécutées sur les clusters principaux.

Les tables et les vues système n'utilisent pas le même modèle de cohérence que les tables standard. Il est important d'être conscient de ce problème lors de leur interrogation, surtout pour les tables STV et les vues SVV. Par exemple, prenons une table standard t1 avec une colonne c1. Vous pourriez vous attendre à ce que la requête suivante ne renvoie aucune ligne :

```
select * from t1
where c1 > (select max(c1) from t1)
```

Toutefois, la requête suivante sur une table système peut très bien renvoyer des lignes :

```
select * from stv_exec_state
where currenttime > (select max(currenttime) from stv_exec_state)
```

La raison pour laquelle cette requête peut renvoyer des lignes est que currenttime est temporaire et que les deux références dans la requête peuvent ne pas renvoyer la même valeur lors de l'évaluation.

En revanche, la requête suivante ne peut très bien ne renvoyer aucune ligne :

```
select * from stv_exec_state
where currenttime = (select max(currenttime) from stv_exec_state)
```

Visibilité des données dans les tables et vues système

Il existe deux classes de visibilité pour les données dans les tables et les vues système : la visibilité des utilisateurs et la visibilité des super-utilisateurs.

Seuls les utilisateurs disposant de privilèges peuvent afficher les données de ces tables qui se trouvent dans la catégorie visible du super-utilisateur. Les utilisateurs standard peuvent afficher les données des tableaux visibles de l'utilisateur. Pour donner à un utilisateur normal l'accès aux tables visibles par les superutilisateurs, accordez le privilège SELECT sur ces tables à l'utilisateur normal. Pour de plus amples informations, veuillez consulter GRANT.

Par défaut, dans la plupart des tableaux visibles des utilisateurs, les lignes générées par un autre utilisateur sont invisibles pour un utilisateur standard. Si un utilisateur normal bénéficie d'un accès SYSLOG ILLIMITÉ, il peut voir toutes les lignes des tables visibles par l'utilisateur, y compris les lignes générées par un autre utilisateur. Pour plus d'informations, consultez ALTER USER ou CREATE USER. Toutes les lignes de SVV_TRANSACTIONS sont visibles de tous les utilisateurs. Pour plus d'informations sur la visibilité des données, consultez l'article de la base de AWS re:Post connaissances Comment autoriser les utilisateurs réguliers de la base de données Amazon Redshift à consulter les données d'autres utilisateurs de mon cluster dans les tables système ? .

Pour les vues de métadonnées, Amazon Redshift n'autorise pas la visibilité aux utilisateurs auxquels un ACCÈS SYSLOG EST ACCORDÉ SANS RESTRICTION.

> **Note**
>
> Accorder à un utilisateur un accès illimité aux tableaux système permet à celui-ci de voir les données générées par d'autres utilisateurs. Par exemple, STL_QUERY et STL_QUERY_TEXT contiennent le texte complet des instructions INSERT, UPDATE et DELETE, qui peuvent contenir des données confidentielles générées par l'utilisateur.

Un super-utilisateur peut afficher toutes les lignes de toutes les tables. Pour permettre à un utilisateur standard d'avoir accès aux tableaux visibles du super-utilisateur, GRANT-lui le privilège SELECT sur ce tableau.

Filtrage des requêtes générées par le système

Les tables et vues système associées à une requête, telles que SVL_QUERY_SUMMARY, SVL_QLOG et d'autres, contiennent généralement un grand nombre d'instructions générées automatiquement qu'Amazon Redshift utilise pour surveiller le statut de la base de données. Ces requêtes générées par le système sont visibles par un super-utilisateur, mais sont rarement utiles. Pour les filtrer lors de la sélection d'une table ou d'une vue système qui utilise la colonne `userid`, ajoutez la condition `userid > 1` à la clause WHERE. Par exemple :

```
select * from svl_query_summary where userid > 1
```

Migration de requêtes provisionnées uniquement vers des requêtes SYS Monitoring View

Migration à partir de clusters provisionnés vers Amazon Redshift sans serveur

Si vous migrez un cluster provisionné vers Amazon Redshift Serverless, vous pouvez avoir des requêtes utilisant les vues système suivantes, qui stockent uniquement les données des clusters provisionnés.

- Toutes les vues STL
- Toutes les vues STV
- Toutes les vues SVCS
- Toutes les vues SVL
- Certaines vues SVV
  - Pour obtenir la liste complète des vues SVV non prises en charge dans Amazon Redshift Serverless, consultez la liste au bas de la section Surveillance des requêtes et des charges de travail avec Amazon Redshift Serverless dans le guide de gestion Amazon Redshift.

Pour continuer à utiliser vos requêtes, réajustez-les pour utiliser les colonnes définies dans les vues de surveillance SYS qui correspondent aux colonnes de vos vues provisionnées uniquement. Pour voir la relation de mappage entre les vues provisionnées uniquement et les vues de surveillance SYS, rendez-vous sur Cartographie des vues système pour la migration vers les vues de surveillance SYS

Mise à jour des requêtes tout en restant sur un cluster provisionné

Si vous ne migrez pas vers Amazon Redshift sans serveur, vous souhaiterez peut-être tout de même mettre à jour vos requêtes existantes. Conçues pour être faciles à utiliser et réduire la complexité, les vues de surveillance SYS fournissent une gamme complète de métriques pour une surveillance et un dépannage efficaces. En utilisant des vues SYS telles que SYS_QUERY_HISTORY et SYS_QUERY_DETAIL qui consolident les informations de plusieurs vues provisionnées uniquement, vous pouvez rationaliser vos requêtes.

Amélioration du suivi des identificateurs de requêtes à l'aide des vues de surveillance SYS

Les vues de surveillance SYS telles que SYS_QUERY_HISTORY et SYS_QUERY_DETAIL comportent la colonne query_id, qui contient l'identifiant des requêtes des utilisateurs. De même, les vues provisionnées uniquement telles que STL_QUERY et SVL_QLOG comportent la colonne de requête, qui contient également les identifiants de requête. Cependant, les identifiants de requête enregistrés dans les vues du système SYS sont différents de ceux enregistrés dans les vues provisionnées uniquement.

La différence entre les valeurs de la colonne query_id des vues SYS et celles de la colonne de requête des vues provisionnées uniquement est la suivante :

- Dans les vues SYS, la colonne query_id enregistre les requêtes soumises par les utilisateurs dans leur forme d'origine. L'optimiseur Amazon Redshift peut les décomposer en requêtes enfant pour améliorer les performances, mais une requête que vous exécutez ne comportera toujours qu'une seule ligne dans SYS_QUERY_HISTORY. Si vous souhaitez consulter les requêtes enfant individuelles, elles sont disponibles dans SYS_QUERY_DETAIL.
- Dans les vues provisionnées uniquement, la colonne de requête enregistre les requêtes au niveau enfant. Si l'optimiseur Amazon Redshift réécrit votre requête d'origine en plusieurs requêtes enfant, il existera plusieurs lignes dans STL_QUERY comportant des valeurs d'identifiant de requête différentes pour une seule requête exécutée.

Quand vous migrez vos requêtes de surveillance et de diagnostic des vues provisionnées uniquement vers des vues SYS, tenez compte de cette différence et modifiez vos requêtes en conséquence. Pour plus d'informations sur la façon dont Amazon Redshift traite les requêtes, consultez Workflow d'exécution et de planification de requête.

exemple

Pour voir comment Amazon Redshift enregistre les requêtes différemment dans les vues provisionnées uniquement et dans les vues de surveillance SYS, consultez l'exemple de requête suivant. Il s'agit d'une requête écrite telle que vous l'exécuteriez dans Amazon Redshift.

```
SELECT
  s_name
  , COUNT(*) AS numwait
FROM
  supplier,
  lineitem l1,
  orders,
  nation
WHERE    s_suppkey = l1.l_suppkey
         AND o_orderkey = l1.l_orderkey
         AND o_orderstatus = 'F'
         AND l1.l_receiptdate > l1.l_commitdate
         AND EXISTS (SELECT
                       *
                     FROM
                       lineitem l2
                     WHERE  l2.l_orderkey = l1.l_orderkey
                            AND l2.l_suppkey <> l1.l_suppkey )
         AND NOT EXISTS (SELECT
                           *
                         FROM
                           lineitem l3
                         WHERE  l3.l_orderkey = l1.l_orderkey
                                AND l3.l_suppkey <> l1.l_suppkey
                                AND l3.l_receiptdate > l3.l_commitdate )
         AND s_nationkey = n_nationkey
         AND n_name = 'UNITED STATES'
GROUP BY
  s_name
ORDER BY
  numwait DESC
```

```
  , s_name LIMIT 100;
```

À l'arrière-plan, l'optimiseur Amazon Redshift réécrit la requête ci-dessus soumise par l'utilisateur en cinq requêtes enfant.

La première requête enfant crée une table temporaire pour matérialiser une sous-requête.

```
CREATE TEMP TABLE volt_tt_606590308b512(l_orderkey
                                       , l_suppkey
                                       , s_name   ) AS SELECT
                                                     l1.l_orderkey
                                                     , l1.l_suppkey
                                                     , public.supplier.s_name
                                                   FROM
                                                     public.lineitem AS l1,
                                                     public.nation,
                                                     public.orders,
                                                     public.supplier
                                                   WHERE  l1.l_commitdate <
 l1.l_receiptdate
                                                          AND l1.l_orderkey =
 public.orders.o_orderkey
                                                          AND l1.l_suppkey =
 public.supplier.s_suppkey
                                                          AND public.nation.n_name
 = 'UNITED STATES'::CHAR(8)
                                                          AND
 public.nation.n_nationkey = public.supplier.s_nationkey
                                                          AND
 public.orders.o_orderstatus = 'F'::CHAR(1);
```

La deuxième requête enfant collecte des statistiques depuis la table temporaire.

```
padb_fetch_sample: select count(*) from volt_tt_606590308b512;
```

La troisième requête enfant crée une autre table temporaire pour matérialiser une autre sous-requête, en référençant la table temporaire créée ci-dessus.

```
CREATE TEMP TABLE volt_tt_606590308c2ef(l_orderkey
                                       , l_suppkey) AS (SELECT

 volt_tt_606590308b512.l_orderkey
```

```
                                                              ,
 volt_tt_606590308b512.l_suppkey
                                                            FROM
                                                              public.lineitem AS l2,
                                                              volt_tt_606590308b512
                                                            WHERE  l2.l_suppkey <>
 volt_tt_606590308b512.l_suppkey
                                                                   AND l2.l_orderkey =
 volt_tt_606590308b512.l_orderkey)
                                                                   EXCEPT distinct (SELECT
 volt_tt_606590308b512.l_orderkey, volt_tt_606590308b512.l_suppkey
                                                                   FROM public.lineitem AS
 l3, volt_tt_606590308b512
                                                                   WHERE l3.l_commitdate <
 l3.l_receiptdate
                                                                     AND l3.l_suppkey <>
 volt_tt_606590308b512.l_suppkey
                                                                     AND l3.l_orderkey =
 volt_tt_606590308b512.l_orderkey);
```

La quatrième requête enfant collecte les statistiques de la table temporaire.

```
padb_fetch_sample: select count(*) from volt_tt_606590308c2ef
```

La dernière requête enfant utilise les tables temporaires créées ci-dessus pour générer la sortie.

```
SELECT
  volt_tt_606590308b512.s_name AS s_name
  , COUNT(*) AS numwait
FROM
  volt_tt_606590308b512,
  volt_tt_606590308c2ef
WHERE    volt_tt_606590308b512.l_orderkey = volt_tt_606590308c2ef.l_orderkey
         AND volt_tt_606590308b512.l_suppkey = volt_tt_606590308c2ef.l_suppkey
GROUP BY
  1
ORDER BY
  2 DESC
  , 1 ASC LIMIT 100;
```

Dans la vue système provisionnée uniquement STL_QUERY, Amazon Redshift enregistre cinq lignes au niveau de la requête enfant, comme suit :

```
SELECT userid, xid, pid, query, querytxt::varchar(100);
FROM stl_query
WHERE xid = 48237350
ORDER BY xid, starttime;

 userid |   xid    |    pid     |  query   |
    querytxt
--------+----------+------------+----------
+-----------------------------------------------------------------------------------------------------------
    101 | 48237350 | 1073840810 | 12058151 | CREATE TEMP TABLE
 volt_tt_606590308b512(l_orderkey, l_suppkey, s_name) AS SELECT l1.l_orderkey, l1.l
    101 | 48237350 | 1073840810 | 12058152 | padb_fetch_sample: select count(*) from
 volt_tt_606590308b512
    101 | 48237350 | 1073840810 | 12058156 | CREATE TEMP TABLE
 volt_tt_606590308c2ef(l_orderkey, l_suppkey) AS (SELECT volt_tt_606590308b512.l_or
    101 | 48237350 | 1073840810 | 12058168 | padb_fetch_sample: select count(*) from
 volt_tt_606590308c2ef
    101 | 48237350 | 1073840810 | 12058170 | SELECT s_name , COUNT(*) AS numwait FROM
 supplier, lineitem l1, orders, nation WHERE s_suppkey = l1.
(5 rows)
```

Dans la vue de surveillance SYS SYS_QUERY_HISTORY, Amazon Redshift enregistre la requête comme suit :

```
SELECT user_id, transaction_id, session_id, query_id, query_text::varchar(100)
FROM sys_query_history
WHERE transaction_id = 48237350
ORDER BY start_time;

 user_id | transaction_id | session_id | query_id |
          query_text
---------+----------------+------------+----------
+-----------------------------------------------------------------------------------------------------------
     101 |       48237350 | 1073840810 | 12058149 | SELECT s_name , COUNT(*) AS numwait
 FROM supplier, lineitem l1, orders, nation WHERE s_suppkey = l1.
```

Dans SYS_QUERY_DETAIL, vous pouvez trouver des information de niveau requête enfant à l'aide de la valeur query_id de SYS_QUERY_HISTORY. La colonne child_query_sequence indique l'ordre dans lequel les requêtes enfant sont exécutées. Pour plus d'informations sur les colonnes de SYS_QUERY_DETAIL, consultez SYS_QUERY_DETAIL.

```
select user_id,
```

```
       query_id,
       child_query_sequence,
       stream_id,
       segment_id,
       step_id,
       start_time,
       end_time,
       duration,
       blocks_read,
       blocks_write,
       local_read_io,
       remote_read_io,
       data_skewness,
       time_skewness,
       is_active,
       spilled_block_local_disk,
       spilled_block_remote_disk
from sys_query_detail
where query_id = 12058149
      and step_id = -1
order by query_id,
         child_query_sequence,
         stream_id,
         segment_id,
         step_id;

 user_id | query_id | child_query_sequence | stream_id | segment_id | step_id |
       start_time         |          end_time          | duration | blocks_read |
 blocks_write | local_read_io | remote_read_io | data_skewness | time_skewness |
 is_active | spilled_block_local_disk | spilled_block_remote_disk
---------+----------+----------------------+-----------+------------+---------
+----------------------------+----------------------------+----------+-------------
+--------------+---------------+----------------+---------------+---------------
+-----------+--------------------------+--------------------------
     101 | 12058149 |                    1 |         0 |          0 |      -1 |
 2023-09-27 15:40:38.512415 | 2023-09-27 15:40:38.533333 |    20918 |           0 |
         0 |             0 |              0 |             0 |            44 | f
 |                        0 |                           0
     101 | 12058149 |                    1 |         1 |          1 |      -1 |
 2023-09-27 15:40:39.931437 | 2023-09-27 15:40:39.972826 |    41389 |          12 |
         0 |            12 |              0 |             0 |            77 | f
 |                        0 |                           0
     101 | 12058149 |                    1 |         2 |          2 |      -1 |
 2023-09-27 15:40:40.584412 | 2023-09-27 15:40:40.613982 |    29570 |          32 |
```

```
        0 |            32 |              0 |             0 |           25 | f
|                        0 |                           0
    101 | 12058149 |                    1 |         2 |          3 |      -1 |
2023-09-27 15:40:40.582038 | 2023-09-27 15:40:40.615758 |    33720 |           0 |
        0 |             0 |              0 |             0 |            1 | f
|                        0 |                           0
    101 | 12058149 |                    1 |         3 |          4 |      -1 |
2023-09-27 15:40:46.668766 | 2023-09-27 15:40:46.705456 |    36690 |          24 |
        0 |            15 |              0 |             0 |           17 | f
|                        0 |                           0
    101 | 12058149 |                    1 |         4 |          5 |      -1 |
2023-09-27 15:40:46.707209 | 2023-09-27 15:40:46.709176 |     1967 |           0 |
        0 |             0 |              0 |             0 |           18 | f
|                        0 |                           0
    101 | 12058149 |                    1 |         4 |          6 |      -1 |
2023-09-27 15:40:46.70656  | 2023-09-27 15:40:46.71289  |     6330 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                        0 |                           0
    101 | 12058149 |                    1 |         5 |          7 |      -1 |
2023-09-27 15:40:46.71405  | 2023-09-27 15:40:46.714343 |      293 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                        0 |                           0
    101 | 12058149 |                    2 |         0 |          0 |      -1 |
2023-09-27 15:40:52.083907 | 2023-09-27 15:40:52.087854 |     3947 |           0 |
        0 |             0 |              0 |             0 |           35 | f
|                        0 |                           0
    101 | 12058149 |                    2 |         1 |          1 |      -1 |
2023-09-27 15:40:52.089632 | 2023-09-27 15:40:52.091129 |     1497 |           0 |
        0 |             0 |              0 |             0 |           11 | f
|                        0 |                           0
    101 | 12058149 |                    2 |         1 |          2 |      -1 |
2023-09-27 15:40:52.089008 | 2023-09-27 15:40:52.091306 |     2298 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         0 |          0 |      -1 |
2023-09-27 15:40:56.882013 | 2023-09-27 15:40:56.897282 |    15269 |           0 |
        0 |             0 |              0 |             0 |           29 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         1 |          1 |      -1 |
2023-09-27 15:40:59.718554 | 2023-09-27 15:40:59.722789 |     4235 |           0 |
        0 |             0 |              0 |             0 |           13 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         2 |          2 |      -1 |
2023-09-27 15:40:59.800382 | 2023-09-27 15:40:59.807388 |     7006 |           0 |
```

```
        0 |             0 |              0 |             0 |           58 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         3 |          3 |      -1 |
2023-09-27 15:41:06.488685 | 2023-09-27 15:41:06.493825 |     5140 |           0 |
        0 |             0 |              0 |             0 |           56 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         3 |          4 |      -1 |
2023-09-27 15:41:06.486206 | 2023-09-27 15:41:06.497756 |    11550 |           0 |
        0 |             0 |              0 |             0 |            2 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         4 |          5 |      -1 |
2023-09-27 15:41:06.499201 | 2023-09-27 15:41:06.500851 |     1650 |           0 |
        0 |             0 |              0 |             0 |           15 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         4 |          6 |      -1 |
2023-09-27 15:41:06.498609 | 2023-09-27 15:41:06.500949 |     2340 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                        0 |                           0
    101 | 12058149 |                    3 |         5 |          7 |      -1 |
2023-09-27 15:41:06.502945 | 2023-09-27 15:41:06.503282 |      337 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                        0 |                           0
    101 | 12058149 |                    4 |         0 |          0 |      -1 |
2023-09-27 15:41:06.62899  | 2023-09-27 15:41:06.631452 |     2462 |           0 |
        0 |             0 |              0 |             0 |           22 | f
|                        0 |                           0
    101 | 12058149 |                    4 |         1 |          1 |      -1 |
2023-09-27 15:41:06.632313 | 2023-09-27 15:41:06.63391  |     1597 |           0 |
        0 |             0 |              0 |             0 |           20 | f
|                        0 |                           0
    101 | 12058149 |                    4 |         1 |          2 |      -1 |
2023-09-27 15:41:06.631726 | 2023-09-27 15:41:06.633813 |     2087 |           0 |
        0 |             0 |              0 |             0 |            0 | f
|                        0 |                           0
    101 | 12058149 |                    5 |         0 |          0 |      -1 |
2023-09-27 15:41:12.571974 | 2023-09-27 15:41:12.584234 |    12260 |           0 |
        0 |             0 |              0 |             0 |           39 | f
|                        0 |                           0
    101 | 12058149 |                    5 |         0 |          1 |      -1 |
2023-09-27 15:41:12.569815 | 2023-09-27 15:41:12.585391 |    15576 |           0 |
        0 |             0 |              0 |             0 |            4 | f
|                        0 |                           0
    101 | 12058149 |                    5 |         1 |          2 |      -1 |
2023-09-27 15:41:13.758513 | 2023-09-27 15:41:13.76401  |     5497 |           0 |
```

```
         0 |             0 |              0 |             0 |            39 | f
 |                        0 |                           0
     101 | 12058149 |                    5 |         1 |          3 |      -1 |
 2023-09-27 15:41:13.749    | 2023-09-27 15:41:13.772987 |    23987 |           0 |
         0 |             0 |              0 |             0 |            32 | f
 |                        0 |                           0
     101 | 12058149 |                    5 |         2 |          4 |      -1 |
 2023-09-27 15:41:13.799526 | 2023-09-27 15:41:13.813506 |    13980 |           0 |
         0 |             0 |              0 |             0 |            62 | f
 |                        0 |                           0
     101 | 12058149 |                    5 |         2 |          5 |      -1 |
 2023-09-27 15:41:13.798823 | 2023-09-27 15:41:13.813651 |    14828 |           0 |
         0 |             0 |              0 |             0 |             0 | f
 |                        0 |                           0
(28 rows)
```

Identifiants de requête, de processus et de session dans la table système

Lorsque vous analysez les identifiants de requête, de processus et de session qui apparaissent dans les tables système, tenez compte des points suivants :

- La valeur de l'identifiant de requête (dans des colonnes telles que `query_id` et`query`) peut être réutilisée au fil du temps.
- La valeur de l'identifiant du processus ou de l'identifiant de session (dans des colonnes telles que `process_id`pid, et`session_id`) peut être réutilisée au fil du temps.
- La valeur de l'identifiant de transaction (dans des colonnes telles que `transaction_id` et`xid`) est unique.

Vues de métadonnées SVV

Les vues SVV sont des vues système dans Amazon Redshift qui contiennent des informations sur les objets de base de données. Notez que même s'ils stockent des informations telles que les autorisations des utilisateurs ou les noms des tables, ils ne sont pas destinés à être associés à des relations créées par l'utilisateur.

> **Note**
>
> Amazon Redshift signale un AVERTISSEMENT, et non une ERREUR, si une réponse de la base de données échoue pour une raison quelconque. Amazon Redshift n'envoie pas de messages ERREUR lorsque vous interrogez des objets dans une unité de partage des données.

Rubriques

- SVV_ACTIVE_CURSORS
- SVV_ALL_COLUMNS
- SVV_ALL_SCHEMAS
- SVV_ALL_TABLES
- SVV_ALTER_TABLE_RECOMMENDATIONS
- SVV_ATTACHED_MASKING_POLICY
- SVV_COLUMNS
- SVV_COLUMN_PRIVILEGES
- SVV_COPY_JOB_INTEGRATIONS
- SVV_DATABASE_PRIVILEGES
- SVV_DATASHARE_PRIVILEGES
- SVV_DATASHARES
- SVV_DATASHARE_CONSUMERS
- SVV_DATASHARE_OBJECTS
- SVV_DEFAULT_PRIVILEGES
- SVV_DISKUSAGE
- SVV_EXTERNAL_COLUMNS
- SVV_EXTERNAL_DATABASES
- SVV_EXTERNAL_PARTITIONS
- SVV_EXTERNAL_SCHEMAS
- SVV_EXTERNAL_TABLES
- SVV_FUNCTION_PRIVILEGES
- SVV_GEOGRAPHY_COLUMNS

- SVV_GEOMETRY_COLUMNS
- SVV_IAM_PRIVILEGES
- SVV_IDENTITY_PROVIDERS
- SVV_INTEGRATION
- SVV_INTEGRATION_TABLE_MAPPAGE
- SVV_INTEGRATION_TABLE_STATE
- SVV_INTERLEAVED_COLUMNS
- SVV_LANGUAGE_PRIVILEGES
- SVV_MASKING_POLICY
- SVV_ML_MODEL_INFO
- SVV_ML_MODEL_PRIVILEGES
- SVV_MV_DEPENDENCY
- SVV_MV_INFO
- SVV_QUERY_INFLIGHT
- SVV_QUERY_STATE
- SVV_REDSHIFT_COLUMNS
- SVV_REDSHIFT_DATABASES
- SVV_REDSHIFT_FUNCTIONS
- SVV_REDSHIFT_SCHEMA_QUOTA
- SVV_REDSHIFT_SCHEMAS
- SVV_REDSHIFT_TABLES
- SVV_RELATION_PRIVILEGES
- SVV_RLS_APPLIED_POLICY
- SVV_RLS_ATTACHED_POLICY
- SVV_RLS_POLICY
- SVV_RLS_RELATION
- SVV_ROLE_GRANTS
- SVV_ROLES
- SVV_SCHEMA_PRIVILEGES
- SVV_SCHEMA_QUOTA_STATE

- SVV_SYSTEM_PRIVILEGES
- SVV_TABLE_INFO
- SVV_TABLES
- SVV_TRANSACTIONS
- SVV_USER_GRANTS
- SVV_USER_INFO
- SVV_VACUUM_PROGRESS
- SVV_VACUUM_SUMMARY

SVV_ACTIVE_CURSORS

STV_ACTIVE_CURSORS affiche des détails concernant les curseurs actuellement ouverts. Pour de plus amples informations, veuillez consulter DECLARE.

SVV_ACTIVE_CURSORS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système. Un utilisateur ne peut voir que les curseurs ouverts par lui-même. Un super-utilisateur peut afficher tous les curseurs.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	ID de l'utilisateur qui a créé le curseur.
nom_curseur	varchar(128)	Nom du curseur.
transaction_id	bigint(128)	ID de la transaction.
session_id	entier	ID du processus avec le curseur actif.
declare_time	timestamp	Heure à laquelle le curseur a été déclaré.

Nom de la colonne	Type de données	Description
total_bytes	bigint	Taille du jeu de résultats du curseur, en octets.
total_rows	bigint	Nombre de lignes dans le jeu de résultats du curseur.
fetched_rows	bigint	Nombre de lignes actuellement extraites du jeu de résultats du curseur.
cursor_storage_limit_used_percent	entier	Pourcentage d'espace disque actuellement utilisé par le curseur.

SVV_ALL_COLUMNS

Utilisez SVV_ALL_COLUMNS pour afficher une union de colonnes des tables Amazon Redshift comme indiqué dans SVV_REDSHIFT_COLUMNS et la liste consolidée des colonnes externes de toutes les tables externes. Pour de plus amples informations sur les colonnes Amazon Redshift, consultez SVV_REDSHIFT_COLUMNS.

SVV_ALL_COLUMNS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	varchar(128)	Nom de la base de données.
nom_schéma	varchar(128)	Nom du schéma.
table_name	varchar(128)	Nom de la table.
column_name	varchar(128)	Le nom de la colonne.

Nom de la colonne	Type de données	Description
ordinal_position	entier	Position de la colonne dans la table.
column_default	varchar(4000)	Valeur par défaut de la colonne.
is_nullable	varchar(3)	Valeur qui indique si la colonne est nullable. Les valeurs possibles sont « oui » et « non ».
data_type	varchar(128)	Type de données de la colonne.
character_maximum_length	entier	Nombre maximal de caractère s dans la colonne.
numeric_precision	entier	Précision numérique.
numeric_scale	entier	Échelle numérique.
remarks	varchar(256)	Remarques.

Exemples de requêtes

L'exemple suivant renvoie la sortie de SVV_ALL_COLUMNS.

```
SELECT *
FROM svv_all_columns
WHERE database_name = 'tickit_db'
    AND TABLE_NAME = 'tickit_sales_redshift'
ORDER BY COLUMN_NAME,
    SCHEMA_NAME
LIMIT 5;

 database_name | schema_name |     table_name        | column_name | ordinal_position
 | column_default | is_nullable | data_type | character_maximum_length |
 numeric_precision | numeric_scale | remarks
```

```
 --------------+------------+----------------------+------------+-----------------
+---------------+------------+----------+-------------------------
+-------------------+---------------+---------
   tickit_db   |    public   | tickit_sales_redshift |    buyerid  |        4         |
               |      NO     |  integer  |                          |         32
  |       0       |
   tickit_db   |    public   | tickit_sales_redshift | commission  |        9         |
               |     YES     |  numeric  |                          |          8
  | 2       |
   tickit_db   |    public   | tickit_sales_redshift |    dateid   |        7         |
               |      NO     |  smallint |                          |         16
  |       0       |
   tickit_db   |    public   | tickit_sales_redshift |   eventid   |        5         |
               |      NO     |  integer  |                          |         32
  |       0       |
   tickit_db   |    public   | tickit_sales_redshift |    listid   |        2         |
               |      NO     |  integer  |                          |         32
  |       0       |
```

SVV_ALL_SCHEMAS

Utilisez SVV_ALL_SCHEMAS pour afficher une union de schémas Amazon Redshift comme indiqué dans SVV_REDSHIFT_SCHEMAS et la liste consolidée des schémas externes de toutes les bases de données. Pour de plus amples informations sur les schémas Amazon Redshift, consultez SVV_REDSHIFT_SCHEMAS.

SVV_ALL_SCHEMAS est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	varchar(128)	Le nom de la base de données où le schéma existe.
nom_schéma	varchar(128)	Nom du schéma.
schema_owner	entier	ID de l'utilisateur du propriétaire du schéma. Pour plus

Nom de la colonne	Type de données	Description
		d'informations sur l'utilisateur IDs, consultezPG_USER_INFO.
schema_type	varchar(128)	Type de schéma. Les valeurs possibles sont les schémas externes, locaux et partagés.
schema_acl	varchar(128)	Chaîne qui définit les autorisations de l'utilisateur ou du groupe d'utilisateurs spécifié pour le schéma.
source_database	varchar(128)	Le nom de la base de données source pour le schéma externe.
schema_option	varchar(256)	Les options du schéma. Il s'agit d'un attribut de schéma externe.

Exemple de requête

L'exemple suivant renvoie la sortie de SVV_ALL_SCHEMAS.

```
SELECT *
FROM svv_all_schemas
WHERE database_name = 'tickit_db'
ORDER BY database_name,
    SCHEMA_NAME;


 database_name |    schema_name     | schema_owner | schema_type | schema_acl |
 source_database | schema_option
---------------+--------------------+--------------+-------------+------------
+-----------------+--------------
```

```
   tickit_db   |       public       |       1      |   shared    |           |
       |
```

SVV_ALL_TABLES

Utilisez SVV_ALL_TABLES pour afficher une union de tables Amazon Redshift comme indiqué dans SVV_REDSHIFT_TABLES et la liste consolidée des tables externes de tous les schémas externes. Pour de plus amples informations sur les tables Amazon Redshift, consultez SVV_REDSHIFT_TABLES.

SVV_ALL_TABLES est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	varchar(128)	Nom de la base de données contenant la table.
nom_schéma	varchar(128)	Nom du schéma de la table.
table_name	varchar(128)	Nom de la table.
table_acl	varchar(128)	Chaîne qui définit l'autorisation de l'utilisateur ou du groupe d'utilisateurs spécifié pour la table.
table_type	varchar(128)	Type de la table. Les valeurs possibles sont les vues, les tables de base, les tables externes et les tables partagées.
remarks	varchar(256)	Remarques.

Exemples de requêtes

L'exemple suivant renvoie la sortie de SVV_ALL_TABLES.

```
SELECT *
FROM svv_all_tables
WHERE database_name = 'tickit_db'
ORDER BY TABLE_NAME,
    SCHEMA_NAME
LIMIT 5;

 database_name | schema_name |        table_name        | table_type | table_acl |
 remarks
---------------+-------------+--------------------------+------------+-----------
+---------
   tickit_db   |    public   | tickit_category_redshift |    TABLE   |           |
   tickit_db   |    public   |   tickit_date_redshift   |    TABLE   |           |
   tickit_db   |    public   |   tickit_event_redshift  |    TABLE   |           |
   tickit_db   |    public   | tickit_listing_redshift  |    TABLE   |           |
   tickit_db   |    public   |   tickit_sales_redshift  |    TABLE   |           |
```

Si la valeur table_acl est nulle, aucun privilège d'accès n'a été explicitement accordé à la table correspondante.

SVV_ALTER_TABLE_RECOMMENDATIONS

Enregistre les recommandations actuelles d'Amazon Redshift Advisor pour les tables. Cette vue affiche des recommandations pour toutes les tables, qu'elles soient définies pour l'optimisation automatique ou non. Pour voir si une table est définie pour l'optimisation automatique, consultez SVV_TABLE_INFO. Les entrées n'apparaissent que pour les tables visibles dans la base de données de la séance en cours. Une fois qu'une recommandation a été appliquée (par Amazon Redshift ou par vous), elle n'apparaît plus dans la vue.

SVV_ALTER_TABLE_RECOMMENDATIONS n'est visible que par les super-utilisateurs. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
type	character (30)	Le type de recommandation. Les valeurs possibles sont distkey et sortkey.
database	character (128)	Nom de la base de données.
table_id	entier	L'identificateur de table.
group_id	entier	Le numéro de groupe d'un ensemble de recommandations. Toutes les recommandations d'un groupe doivent être appliquées pour en tirer le maximum d'avantages. Les valeurs possibles sont -1 pour une recommandation de clé de tri et un nombre supérieur à zéro pour une recommandation de clé de distribution.
ddl	character (1024)	Instruction SQL qui doit être exécutée pour appliquer la recommand ation.
auto_eligible	character (1)	La valeur indique si la recommandation est éligible à l'exécution automatique d'Amazon Redshift. Si cette valeur est `t`, alors l'indication est vraie. Parallèlement, si la valeur est `f`, l'indication est fausse.

Exemples de requêtes

Dans l'exemple suivant, les lignes de résultats affichent des recommandations pour la clé de distribution et la clé de tri. Les lignes indiquent également si les recommandations sont éligibles à l'exécution automatique d'Amazon Redshift.

```
select type, database, table_id, group_id, ddl, auto_eligible
from svv_alter_table_recommendations;
```

```
type      | database | table_id | group_id | ddl

                  | auto_eligible
diststyle | db0      | 117884   | 2        | ALTER TABLE "sch"."dp21235_tbl_1" ALTER
DISTSTYLE KEY DISTKEY "c0"
                    | f
diststyle | db0      | 117892   | 2        | ALTER TABLE "sch"."dp21235_tbl_1" ALTER
DISTSTYLE KEY DISTKEY "c0"
                    | f
diststyle | db0      | 117885   | 1        | ALTER TABLE "sch"."catalog_returns"
ALTER DISTSTYLE KEY DISTKEY "cr_sold_date_sk", ALTER COMPOUND SORTKEY
("cr_sold_date_sk","cr_returned_time_sk") | t
sortkey   | db0      | 117890   | -1       | ALTER TABLE "sch"."customer_addresses"
ALTER COMPOUND SORTKEY ("ca_address_sk")
                     | t
```

SVV_ATTACHED_MASKING_POLICY

Utilisez SVV_ATTACHED_MASKING_POLICY pour afficher toutes les relations et tous les rôles/utilisateurs qui disposent de politiques attachées à la base de données actuellement connectée.

Seuls les super-utilisateurs et les utilisateurs ayant le rôle `sys:secadmin` peuvent afficher SVV_ATTACHED_MASKING_POLICY. Les utilisateurs réguliers verront 0 ligne.

Colonnes de la table

Nom de la colonne	Type de données	Description
policy_name	text	Nom de la politique de masquage attachée à la table.
nom_schéma	text	Schéma de la table à laquelle la politique est attachée.
table_name	text	Nom de la table à laquelle la politique est attachée.
table_type	text	Type de la table à laquelle la politique est attachée.

Nom de la colonne	Type de données	Description
grantor	text	Nom de l'utilisateur qui a attaché la politique.
grantee	text	Nom de l'utilisateur/du rôle auquel la politique est attachée.
grantee_type	text	Type de bénéficiaire. Il peut s'agir d'un rôle, d'un utilisateur ou d'un public.
priority	int	Priorité de la politique attachée.
input_columns	text	Attributs de la colonne d'entrée de la politique attachée.
output_columns	text	Attributs de la colonne de sortie de la politique attachée.
is_masking_datashare_on	boolean	Si la table à laquelle la politique est attachée est protégée par DDM sur les partages de données.

Fonctions internes

SVV_ATTACHED_MASKING_POLICY prend en charge les fonctions internes suivantes :

mask_get_policy_for_role_on_column

Obtenez la politique de priorité la plus élevée qui s'applique à une paire colonne/rôle donnée.

Syntaxe

```
mask_get_policy_for_role_on_column
```

```
                        (relschema,
                        relname,
                        colname,
                        rolename);
```

Paramètres

relschema

Nom du schéma dans lequel la politique se trouve.

relname

Nom de la table dans laquelle la politique se trouve.

colname

Nom de la colonne à laquelle la politique est attachée.

rolename

Nom du rôle auquel la politique est attachée.

mask_get_policy_for_user_on_column

Obtenez la politique de priorité la plus élevée qui s'applique à une paire colonne/utilisateur donnée.

Syntaxe

```
mask_get_policy_for_user_on_column
                        (relschema,
                        relname,
                        colname,
                        username);
```

Paramètres

relschema

Nom du schéma dans lequel la politique se trouve.

relname

Nom de la table dans laquelle la politique se trouve.

colname

Nom de la colonne à laquelle la politique est attachée.

rolename

Nom de l'utilisateur auquel la politique est attachée.

SVV_COLUMNS

Utilisez SVV_COLUMNS pour afficher des informations du catalogue sur les colonnes des tables et des vues locales et externes, y compris les vues à liaison tardive.

SVV_COLUMNS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

La vue SVV_COLUMNS associe les métadonnées des tables à partir de Tables catalogue système (tables avec préfixe PG) et de la vue système SVV_EXTERNAL_COLUMNS. Les tables catalogue système décrivent les tables de base de données Amazon Redshift. SVV_EXTERNAL_COLUMNS décrit les tables externes qui sont utilisées avec Amazon Redshift Spectrum.

Tous les utilisateurs peuvent consulter la totalité des lignes des tables catalogue système. Les utilisateurs standard peuvent consulter les définitions de colonnes à partir de la vue SVV_EXTERNAL_COLUMNS uniquement pour les tables externes auquel ils ont accès. Bien que les utilisateurs standard puissent voir les métadonnées des tables de catalogues système, ils ne peuvent sélectionner les données des tables définies par l'utilisateur que s'ils en sont propriétaires ou si l'accès approprié leur a été accordé.

Colonnes de la table

Nom de la colonne	Type de données	Description
table_catalog	text	Nom du catalogue contenant la table.
table_schema	text	Nom du schéma de la table.
table_name	text	Nom de la table.

Nom de la colonne	Type de données	Description
column_name	text	Nom de la colonne.
ordinal_position	int	Position de la colonne dans la table.
column_default	text	Valeur par défaut de la colonne.
is_nullable	text	Valeur qui indique si la colonne est nullable.
data_type	text	Type de données de la colonne.
character_maximum_length	int	Nombre maximal de caractères dans la colonne.
numeric_precision	int	Précision numérique. Si la colonne data_type est numérique, elle renvoie le nombre de chiffres significatifs dans la valeur complète.
numeric_precision_radix	int	Base de précision numérique. Si la colonne data_type est numérique, elle renvoie la base des colonnes numeric_precision et numeric_scale.
numeric_scale	int	Échelle numérique. Si la colonne data_type est numérique, elle renvoie le nombre de chiffres significatifs dans la valeur décimale.
datetime_precision	int	Précision de datetime.

Nom de la colonne	Type de données	Description
interval_type	text	Type d'intervalle.
interval_precision	text	Précision de l'intervalle.
character_set_catalog	text	Catalogue de jeux de caractères.
character_set_schema	text	Schéma de jeu de caractères.
character_set_name	text	Nom du jeu de caractères.
collation_catalog	text	Catalogue de classement.
collation_schema	text	Schéma de classement.
collation_name	text	Nom du classement.
domain_name	text	Nom du domaine.
remarks	text	Remarques.

SVV_COLUMN_PRIVILEGES

Utilisez SVV_COLUMN_PRIVILEGES pour afficher les autorisations de colonne explicitement accordées aux utilisateurs, aux rôles et aux groupes de la base de données actuelle.

SVV_COLUMN_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les identités auxquelles ils ont accès ou qu'ils possèdent.

Colonnes de la table

Nom de la colonne	Type de données	Description
namespace_name	text	Nom de l'espace de noms où existe une relation spécifiée.
relation_name	text	Nom de la relation.
column_name	text	Le nom de la colonne.
privilege_type	text	Type de l'autorisation. Les valeurs possibles sont SELECT ou UPDATE.
identity_id	entier	ID de l'identité. Les valeurs possibles sont ID d'utilisateur, ID de rôle ou ID de groupe.
identity_name	text	Nom de l'identité.
identity_type	text	Type d'identité. Les valeurs possibles sont utilisateur, rôle, groupe ou public.

Exemple de requête

L'exemple suivant montre le résultat de SVV_COLUMN_PRIVILEGES.

```
SELECT
 namespace_name,relation_name,COLUMN_NAME,privilege_type,identity_name,identity_type
FROM svv_column_privileges WHERE relation_name = 'lineitem';

 namespace_name | relation_name | column_name | privilege_type | identity_name |
 identity_type
----------------+---------------+-------------+----------------+---------------
+----------------
    public      |   lineitem    | l_orderkey  |     SELECT     |    reguser    |
 user
```

```
   public      |   lineitem    | l_orderkey  |     SELECT     |     role1     |
role
   public      |   lineitem    | l_partkey   |     SELECT     |    reguser    |
user
   public      |   lineitem    | l_partkey   |     SELECT     |     role1     |
role
```

SVV_COPY_JOB_INTEGRATIONS

Utilisez SVV_COPY_JOB_INTEGRATIONS pour afficher les détails des intégrations d'événements S3.

Cette vue contient les intégrations d'événements S3 qui ont été créées.

SVV_COPY_JOB_INTEGRATIONS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
propriétaire du poste	entier	Identifiant du propriétaire de la tâche.
channel_arn	name	Identifiant d'intégration.
bucket	text	Le nom du compartiment Amazon S3 associé à l'intégration.
état du canal	name	État de l’intégration. Valeurs valides : `PendingEstablished` , et`Inactive`.
db_name	name	Nom de base de données de l'objet dépendant.
job_name	text	Nom de la tâche .

Nom de la colonne	Type de données	Description
état_emploi	entier	L'état du poste. Valeurs valides : 0 pour actif, 1 pour en attente.

L'exemple suivant renvoie les intégrations S3 pour la base de données actuelle.

```
SELECT * FROM SVV_COPY_JOB_INTEGRATIONS WHERE db_name = pg_catalog.current_database();
```

SVV_DATABASE_PRIVILEGES

Utilisez SVV_DATABASE_PRIVILEGES pour afficher les autorisations de base de données explicitement accordées aux utilisateurs, aux rôles et aux groupes de votre cluster Amazon Redshift.

SVV_DATABASE_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les identités auxquelles ils ont accès ou qu'ils possèdent.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	text	Nom de la base de données.
privilege_type	text	Type de l'autorisation. Pour les autorisations associées à un privilege_scope de base de données, les valeurs possibles sont USAGE, CREATE, TEMPORARY, TEMP et ALTER. Pour les valeurs privilege_scope autres que celles de base de données, les valeurs possibles

Nom de la colonne	Type de données	Description
		incluent tout type d'autorisation disponible dans le champ d'application de l'autorisation.
identity_id	entier	ID de l’identité. Les valeurs possibles sont ID d’utilisateur, ID de rôle ou ID de groupe.
identity_name	text	Nom de l’identité.
identity_type	text	Type d’identité. Les valeurs possibles sont utilisateur, rôle, groupe ou public.
admin_option	boolean	Valeur qui indique si l’utilisateur peut accorder l’autorisation à d’autres utilisateurs et rôles. Elle renvoie toujours une réponse fausse pour les types d’identités rôle et groupe.
privilege_scope	text	Étendue de l'autorisation spécifiée dans privilege_type. Les valeurs possibles sont les suivantes : • DATABASE • SCHEMAS • TABLES • FONCTIONS • LANGAGES Pour plus d'informations sur les autorisations délimitées, rendez-vous sur. Autorisations étendues

Exemple de requête

L’exemple suivant montre le résultat de SVV_DATABASE_PRIVILEGES.

```
SELECT database_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_database_privileges
```

```
WHERE database_name = 'test_db';

 database_name | privilege_type | identity_name | identity_type | admin_option
---------------+----------------+---------------+---------------+--------------
     test_db   |     CREATE     |     reguser   |      user     |     False
     test_db   |     CREATE     |      role1    |      role     |     False
     test_db   |     TEMP       |      public   |      public   |     False
     test_db   |     TEMP       |      role1    |      role     |     False
```

SVV_DATASHARE_PRIVILEGES

Utilisez SVV_DATASHARE_PRIVILEGES pour afficher les autorisations d'unité de partage des données explicitement accordées aux utilisateurs, aux rôles et aux groupes de votre cluster Amazon Redshift.

SVV_DATASHARE_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l’autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les identités auxquelles ils ont accès ou qu’ils possèdent.

Colonnes de la table

Nom de la colonne	Type de données	Description
datashare_name	text	Nom de l’unité de partage des données.
privilege_type	text	Type de l’autorisation. Les valeurs possibles sont ALTER ou SHARE.
identity_id	entier	ID de l’identité. Les valeurs possibles sont ID d’utilisateur, ID de rôle ou ID de groupe.
identity_name	text	Nom de l’identité.

Nom de la colonne	Type de données	Description
identity_type	text	Type d'identité. Les valeurs possibles sont utilisateur, rôle, groupe ou public.
admin_option	boolean	Valeur qui indique si l'utilisateur peut accorder l'autorisation à d'autres utilisateurs et rôles. Elle renvoie toujours une réponse fausse pour les types d'identités rôle et groupe.

Exemple de requête

L'exemple suivant montre le résultat de SVV_DATASHARE_PRIVILEGES.

```
SELECT datashare_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_datashare_privileges
WHERE datashare_name = 'demo_share';

 datashare_name | privilege_type |  identity_name | identity_type | admin_option
----------------+----------------+----------------+---------------+--------------
   demo_share   |     ALTER      |    superuser   |     user      |   False
   demo_share   |     ALTER      |    reguser     |     user      |   False
```

SVV_DATASHARES

Utilisez SVV_DATASHARES pour afficher la liste des unités de partage des données créées sur le cluster et de celles partagées avec le cluster.

SVV_DATASHARES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Propriétaire de l'unité de partage des données
- Utilisateurs disposant des autorisations ALTER ou USAGE sur une unité de partage des données

Les autres utilisateurs ne peuvent afficher les lignes. Pour en savoir plus sur les autorisations ALTER et USAGE, consultez GRANT.

Colonnes de la table

Nom de la colonne	Type de données	Description
share_name	varchar(128)	Le nom d'une unité de partage des données.
share_id	entier	L'ID de l'unité de partage des données.
share_owner	entier	Le propriétaire de l'unité de partage des données.
source_database	varchar(128)	La base de données source de cette unité de partage des données.
consumer_database	varchar(128)	La base de données consommateur créée à partir de cette unité de partage des données.
share_type	varchar(8)	Le type d'unité de partage des données. Les valeurs possibles sont INBOUND et OUTBOUND.
createdate	horodatage sans fuseau horaire	La date de création de l'unité de partage des données.
is_publicaccessible	boolean	La propriété spécifiant si une unité de partage des données peut être partagée avec un cluster accessible au public.
share_acl	varchar(256)	Chaîne qui définit les autorisations de l'utilisateur ou du groupe d'utilisateurs spécifié

Nom de la colonne	Type de données	Description
		pour l'unité de partage des données.
producer_account	varchar(16)	L'ID du compte producteu r de l'unité de partage des données.
producer_namespace	varchar(64)	L'ID unique du cluster producteur de l'unité de partage des données.
managed_by	varchar(64)	Propriété qui indique le AWS service qui gère le partage de données.

Notes d'utilisation

Récupération de métadonnées supplémentaires : à l'aide de l'entier renvoyé dans la `share_owner` colonne, vous pouvez vous joindre `usesysid` SVL_USER_INFO à la colonne pour obtenir des données sur le propriétaire du partage de données. Cela inclut le nom et les propriétés supplémentaires.

Exemple de requête

L'exemple suivant renvoie la sortie pour SVV_DATASHARES.

```
SELECT share_owner, source_database, share_type, is_publicaccessible
FROM svv_datashares
WHERE share_name LIKE 'tickit_datashare%'
AND source_database = 'dev';

  share_owner | source_database | share_type  | is_publicaccessible
--------------+-----------------+-------------+---------------------
     100      |      dev        |   OUTBOUND  |        True
(1 rows)
```

L'exemple suivant renvoie la sortie pour SVV_DATASHARES pour des unités de partage des données sortantes.

```
SELECT share_name, share_owner, btrim(source_database), btrim(consumer_database),
 share_type, is_publicaccessible, share_acl, btrim(producer_account),
 btrim(producer_namespace), btrim(managed_by) FROM svv_datashares WHERE share_type =
 'OUTBOUND';

   share_name   | share_owner | source_database | consumer_database | share_type |
 is_publicaccessible | share_acl | producer_account|         producer_namespace
   | managed_by
----------------+-------------+-----------------+-------------------
+------------+---------------------+-----------+-----------------
+--------------------------------------+------------
    salesshare  |      1      |       dev       |                   |  OUTBOUND  |
   True         |           |   123456789012  | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |

 marketingshare |      1      |       dev       |                   |  OUTBOUND  |
   True         |           |   123456789012  | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |
```

L'exemple suivant renvoie la sortie pour SVV_DATASHARES pour des unités de partage des données entrantes.

```
SELECT share_name, share_owner, btrim(source_database), btrim(consumer_database),
 share_type, is_publicaccessible, share_acl, btrim(producer_account),
 btrim(producer_namespace), btrim(managed_by) FROM svv_datashares WHERE share_type =
 'INBOUND';

  share_name    | share_owner | source_database | consumer_database | share_type |
 is_publicaccessible | share_acl | producer_account |         producer_namespace
    | managed_by
----------------+-------------+-----------------+-------------------
+------------+---------------------+-----------+-----------------
+--------------------------------------+------------
  salesshare    |             |                 |                   |  INBOUND   |
   False         |           |  123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d
 |
 marketingshare |             |                 |                   |  INBOUND   |
  False         |           |  123456789012    | 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |
 ADX
```

SVV_DATASHARE_CONSUMERS

Utilisez SVV_DATASHARE_CONSUMERS pour afficher la liste des consommateurs de l'unité de partage des données créée sur un cluster.

SVV_DATASHARE_CONSUMERS est visible par les utilisateurs suivants :

- Super-utilisateurs
- Propriétaire de l'unité de partage des données
- Utilisateurs disposant des autorisations ALTER ou USAGE sur une unité de partage des données

Les autres utilisateurs ne peuvent afficher les lignes. Pour en savoir plus sur les autorisations ALTER et USAGE, consultez GRANT.

Colonnes de la table

Nom de la colonne	Type de données	Description
share_name	varchar(128)	Le nom du datashare.
consumer_account	varchar(16)	L'ID du compte consommateur de l'unité de partage des données.
consumer_namespace	varchar(64)	L'ID unique du cluster consommateur de l'unité de partage des données.
share_date	horodatage sans fuseau horaire	La date à laquelle l'unité de partage des données a été partagée.

Exemple de requête

L'exemple suivant renvoie la sortie pour SVV_DATASHARE_CONSUMERS.

```
SELECT count(*)
FROM svv_datashare_consumers
```

```
WHERE share_name LIKE 'tickit_datashare%';

1
```

SVV_DATASHARE_OBJECTS

Utilisez SVV_DATASHARE_OBJECTS pour afficher la liste d'objets des unités de partage des données créés sur le cluster ou partagés avec le cluster.

SVV_DATASHARE_OBJECTS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Pour en savoir plus sur l'affichage d'une liste des unités de partage des données, consultez SVV_DATASHARES.

Colonnes de la table

Nom de la colonne	Type de données	Description
share_type	varchar(8)	Le type de l'unité de partage des données spécifiée. Les valeurs possibles sont OUTBOUND et INBOUND.
share_name	varchar(128)	Le nom du datashare.
object_type	varchar(64)	Le type de l'objet spécifié. Les valeurs possibles sont les schémas, les tables, les vues, les vues à liaison tardive, les vues matérialisées et les fonctions.
nom d'objet	varchar(512)	Nom de l'objet. Le nom de l'objet s'allonge pour inclure le nom du schéma, tel que schema1.t1.

Nom de la colonne	Type de données	Description
producer_account	varchar(16)	L'ID du compte producteur de l'unité de partage des données.
producer_namespace	varchar(64)	L'ID unique du cluster producteur de l'unité de partage des données.
include_new	boolean	Propriété qui indique s'il faut ajouter au partage de données les futures tables, vues ou fonctions SQL définies par l'utilisateur (UDFs) créées dans le schéma spécifié. Ce paramètre n'est pertinent que pour les jeux de données OUTBOUND et uniquement pour les types de schéma de l'unité de partage des données.

Exemple de requête

Les exemples suivants renvoient la sortie pour SVV_DATASHARE_OBJECTS.

```
SELECT share_type,
    btrim(share_name)::varchar(16) AS share_name,
    object_type,
    object_name
FROM svv_datashare_objects
WHERE share_name LIKE 'tickit_datashare%'
AND object_name LIKE '%tickit%'
ORDER BY object_name
LIMIT 5;

 share_type |     share_name     | object_type |          object_name
------------+--------------------+-------------+--------------------------------
```

```
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_category_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_date_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_event_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_listing_redshift
 OUTBOUND   |  tickit_datashare  |    table    |  public.tickit_sales_redshift
```

```
SELECT * FROM SVV_DATASHARE_OBJECTS WHERE share_name like 'sales%';

share_type | share_name | object_type | object_name  | producer_account |
 producer_namespace          | include_new
-----------+------------+-------------+--------------+-----------------
+--------------------------------------+-------------
 OUTBOUND  | salesshare | schema      | public       | 123456789012     |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |      t
 OUTBOUND  | salesshare | table       | public.sales | 123456789012     |
 13b8833d-17c6-4f16-8fe4-1a018f5ed00d |
```

SVV_DEFAULT_PRIVILEGES

Utilisez SVV_DEFAULT_PRIVILEGES pour afficher les privilèges par défaut auxquels un utilisateur a accès dans un cluster Amazon Redshift.

SVV_DEFAULT_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l’autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les autorisations par défaut qui leur sont accordées.

Colonnes de la table

Nom de la colonne	Type de données	Description
nom_schéma	text	Nom du schéma.
object_type	text	Le type d’objet. Les valeurs possibles sont RELATION, FONCTION ou PROCEDURE.

Nom de la colonne	Type de données	Description
owner_id	entier	L'ID du propriétaire. La valeur possible est l'ID utilisateur.
owner_name	text	Le nom du propriétaire.
owner_type	text	Type de propriétaire. La valeur possible est l'utilisateur.
privilege_type	text	Le type de privilège. Les valeurs possibles sont INSERT, SELECT, UPDATE, DELETE, RULE, REFERENCES TRIGGER, DROP et EXECUTE.
grantee_id	entier	L'ID du bénéficiaire. Les valeurs possibles sont ID d'utilisateur, ID de rôle et ID de groupe.
grantee_type	text	Le type de bénéficiaire. Les valeurs possibles sont utilisateur, rôle et public.
admin_option	boolean	La valeur qui indique si l'utilisateur peut accorder des autorisations à d'autres utilisateurs et rôles. Elle est toujours réglée sur false (faux) pour le rôle et le type de groupe.

Exemple de requête

L'exemple suivant renvoie la sortie pour SVV_DEFAULT_PRIVILEGES.

```
SELECT * from svv_default_privileges;

 schema_name |    object_type    | owner_id | owner_name | owner_type | privilege_type
 | grantee_id | grantee_name | grantee_type | admin_option
-------------+-------------------+--------- +------------+------------+---------------
+------------+--------------+--------------+-------------+
   public    |     RELATION      |    106   |     u1     |    user    |     UPDATE
 |     107    |      u2      |     user     |      f      |
   public    |     RELATION      |    106   |     u1     |    user    |     SELECT
 |     107    |      u2      |     user     |      f      |
```

SVV_DISKUSAGE

Amazon Redshift crée la vue système SVV_DISKUSAGE en joignant les tables STV_TBL_PERM et STV_BLOCKLIST. La vue SVV_DISKUSAGE contient des informations sur l'allocation des données pour les tables d'une base de données.

Utilisez les requêtes d'agrégation avec SVV_DISKUSAGE, comme dans les exemples suivants, afin de déterminer le nombre de blocs de disque alloués par base de données, table, coupe ou colonne. Chaque bloc de données utilise 1 Mo. Vous pouvez également utiliser STV_PARTITIONS afin d'afficher des informations de synthèse sur l'utilisation du disque.

SVV_DISKUSAGE n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> Cette vue n'est disponible que lors de l'interrogation des clusters alloués.

Colonnes de la table

Nom de la colonne	Type de données	Description
db_id	entier	ID de base de données.
name	character (72)	Nom de la table.
slice	entier	Tranche de données allouée à la table.
col	entier	Index de base zéro de la colonne. Chaque table que vous créez possède trois colonnes masquées qui lui sont ajoutées : INSERT_XID, DELETE_XID et ROW_ID (OID). Une table avec 3 colonnes définies par l'utilisateur contient 6 colonnes réelles et les colonnes définies par l'utilisateur sont numérotées en interne 0, 1 et 2. Les colonnes INSERT_XID, DELETE_XID et ROW_ID sont numérotés respectivement 3, 4 et 5 dans cet exemple.

Nom de la colonne	Type de données	Description
tbl	entier	ID de table.
blocknum	entier	ID du bloc de données.
num_values	entier	Nombre de valeurs contenues dans le bloc.
minvalue	bigint	Valeur minimale contenue dans le bloc.
maxvalue	bigint	Valeur maximale contenue dans le bloc.
sb_pos	entier	Identificateur interne de la position du super bloc sur le disque.
pinned	entier	Indique si le bloc est mis en mémoire dans le cadre du préchargement ou non. 0 = faux ; 1 = vrai. La valeur par défaut est false.
on_disk	entier	Indique si le bloc est automatiquement stocké sur le disque ou non. 0 = faux ; 1 = vrai. La valeur par défaut est false.
modifié	entier	Indique si le bloc a été modifié ou non. 0 = faux ; 1 = vrai. La valeur par défaut est false.
hdr_modified	entier	Indique si l'en-tête de bloc a été modifié ou non. 0 = faux ; 1 = vrai. La valeur par défaut est false.
unsorted	entier	Indique si un bloc est trié ou non. 0 = faux ; 1 = vrai. La valeur par défaut est true.
tombstone	entier	Pour utilisation interne.
preferred_diskno	entier	Numéro de disque sur lequel le bloc doit être, sauf si le disque a échoué. Une fois que le disque a été corrigé, le bloc renvoie sur ce disque.
temporary	entier	Indique si le bloc contient ou non des données temporaires, comme une table temporaire ou les résultats intermédiaires des requêtes. 0 = faux ; 1 = vrai. La valeur par défaut est false.

Nom de la colonne	Type de données	Description
newblock	entier	Indique si un bloc est nouveau (true) ou n'a jamais été validé sur le disque (false). 0 = faux ; 1 = vrai.

Exemples de requêtes

SVV_DISKUSAGE contenant une ligne par bloc de disque alloué, une requête qui sélectionne toutes les lignes renvoie potentiellement un très grand nombre de lignes. Nous recommandons d'utiliser uniquement les requêtes d'agrégation avec SVV_DISKUSAGE.

renvoiez le plus grand nombre de blocs jamais alloués à la colonne 6 de la table USERS (colonne EMAIL) :

```
select db_id, trim(name) as tablename, max(blocknum)
from svv_diskusage
where name='users' and col=6
group by db_id, name;

db_id  | tablename | max
--------+-----------+-----
175857 | users     |   2
(1 row)
```

La requête suivante renvoie des résultats semblables pour toutes les colonnes de la grande table à 10 colonnes appelée SALESNEW. (Les trois dernières lignes des colonnes 10 à 12 correspondent aux colonnes de métadonnées masquées.)

```
select db_id, trim(name) as tablename, col, tbl, max(blocknum)
from svv_diskusage
where name='salesnew'
group by db_id, name, col, tbl
order by db_id, name, col, tbl;

db_id  | tablename  | col |  tbl   | max
--------+------------+-----+--------+-----
175857 | salesnew   |   0 | 187605 | 154
175857 | salesnew   |   1 | 187605 | 154
175857 | salesnew   |   2 | 187605 | 154
```

```
175857 | salesnew   |   3 | 187605 | 154
175857 | salesnew   |   4 | 187605 | 154
175857 | salesnew   |   5 | 187605 |  79
175857 | salesnew   |   6 | 187605 |  79
175857 | salesnew   |   7 | 187605 | 302
175857 | salesnew   |   8 | 187605 | 302
175857 | salesnew   |   9 | 187605 | 302
175857 | salesnew   |  10 | 187605 |   3
175857 | salesnew   |  11 | 187605 |   2
175857 | salesnew   |  12 | 187605 | 296
(13 rows)
```

SVV_EXTERNAL_COLUMNS

Utilisez SVV_EXTERNAL_COLUMNS pour afficher les détails relatifs aux colonnes des tables externes. Utilisez également SVV_EXTERNAL_COLUMNS pour les requêtes entre bases de données afin d'afficher les détails des colonnes de la table sur les bases non connectées auxquelles les utilisateurs ont accès.

SVV_EXTERNAL_COLUMNS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
redshift_database_name	text	Nom de la base de données Amazon Redshift locale.
schemaname	text	Nom du schéma externe Amazon Redshift relatif à la table externe.
tablename	text	Nom de la table externe.
columnname	text	Nom de la colonne.
external_type	text	Type de données de la colonne.

Nom de la colonne	Type de données	Description
columnnum	entier	Nombre de colonnes externes, à partir de 1.
part_key	entier	Ordre de la clé, si la colonne est une clé de partition. La valeur est **0** si la colonne n'est pas une partition.
is_nullable	text	Définit si une colonne peut avoir la valeur null ou non. Certaines valeurs sont `true`, `false` ou " " (chaîne vide) qui ne représente aucune information.

SVV_EXTERNAL_DATABASES

Utilisez SVV_EXTERNAL_DATABASES pour afficher les détails relatifs aux bases de données externes.

SVV_EXTERNAL_DATABASES est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
eskind	entier	Type du catalogue externe pour la base de données ; **1** correspond à un catalogue de données, **2** à un metastore Hive.

Nom de la colonne	Type de données	Description
esoptions	text	Détails du catalogue où réside la base de données.
databasename	text	Nom de la base de données dans le catalogue externe.
location	text	Emplacement de la base de données.
parameters	text	Paramètres de la base de données.

SVV_EXTERNAL_PARTITIONS

Utilisez SVV_EXTERNAL_PARTITIONS pour afficher les détails relatifs aux partitions des tables externes.

SVV_EXTERNAL_PARTITIONS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système..

Colonnes de la table

Nom de la colonne	Type de données	Description
schemaname	text	Nom du schéma externe Amazon Redshift relatif à la table externe avec les partitions spécifiées.
tablename	text	Nom de la table externe.
values	text	Valeurs pour la partition.
location	text	Emplacement de la partition. La taille de la colonne est limitée à 128 caractères. Les valeurs plus longues sont tronquées.
input_format	text	Format d'entrée.

Nom de la colonne	Type de données	Description
output_format	text	Format de sortie.
serialization_lib	text	Bibliothèque de sérialisation.
serde_parameters	text	SerDe paramètres.
compressed	entier	Valeur qui indique si la partition est compressée ; **1** signifie qu'elle est compressée, **0** qu'elle ne l'est pas.
parameters	text	Propriétés de la partition.

SVV_EXTERNAL_SCHEMAS

Utilisez SVV_EXTERNAL_SCHEMAS pour afficher des informations relatives aux schémas externes. Pour plus d'informations, consultez CREATE EXTERNAL SCHEMA.

SVV_EXTERNAL_SCHEMAS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
esoid	oid	ID de schéma externe.
eskind	smallint	Le type de catalogue externe pour le schéma externe : 1 indique un catalogue de données, 2 indique un métastore Hive, 3 indique une requête fédérée pour Aurora PostgreSQL ou Amazon RDS PostgreSQL, 4 indique un schéma pour une base de données Amazon Redshift locale, 5 indique un schéma pour une base de données Amazon Redshift distante, 6 indique un schéma pour une

Nom de la colonne	Type de données	Description
		table système, 8 indique un schéma pour les bases de données MySQL distantes, 9 indique un schéma pour un flux de données Amazon Kinesis et 10 indique un flux de données Amazon Managed Streaming for Apache Kafka.
schemaname	name	External schema name.
esowner	entier	ID d'utilisateur du propriétaire du schéma externe.
databasename	text	Nom de la base de données externe.
esoptions	text	Options de schéma externe.

exemple

L'exemple suivant illustre les détails relatifs aux schémas externes.

```
select * from svv_external_schemas;

esoid  | eskind | schemaname | esowner | databasename | esoptions

-------+--------+------------+---------+--------------
+-------------------------------------------------------------
100133 |      1 | spectrum   |     100 | redshift     |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
```

SVV_EXTERNAL_TABLES

Utilisez SVV_EXTERNAL_TABLES pour afficher les détails relatifs aux tables externes. Pour plus d'informations, consultez CREATE EXTERNAL SCHEMA. Utilisez également SVV_EXTERNAL_TABLES pour les requêtes entre bases de données afin d'afficher les métadonnées sur toutes les tables des bases non connectées auxquelles les utilisateurs ont accès.

SVV_EXTERNAL_TABLES est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
redshift_database_name	text	Nom de la base de données Amazon Redshift locale.
schemaname	text	Nom du schéma externe Amazon Redshift relatif à la table externe.
tablename	text	Nom de la table externe.
tabletype	text	Type de la table. Certaines valeurs sont TABLE, VIEW, MATERIALISED VIEW ou " " (chaîne vide) qui ne représente aucune informati on.
location	text	Emplacement de la table.
input_format	text	Format d'entrée.
output_format	text	Format de sortie.
serialization_lib	text	Bibliothèque de sérialisation.
serde_parameters	text	SerDe paramètres.
compressed	entier	Valeur qui indique si la table est compressée ; **1** signifie qu'elle est compressée, **0** qu'elle ne l'est pas.
parameters	text	Propriétés de la table.

exemple

L'exemple suivant montre les détails svv_external_tables avec un prédicat sur le schéma externe utilisé par une requête fédérée.

```
select schemaname, tablename from svv_external_tables where schemaname = 'apg_tpch';
schemaname  | tablename
------------+-----------
apg_tpch    | customer
apg_tpch    | lineitem
apg_tpch    | nation
apg_tpch    | orders
apg_tpch    | part
apg_tpch    | partsupp
apg_tpch    | region
apg_tpch    | supplier
(8 rows)
```

SVV_FUNCTION_PRIVILEGES

Utilisez SVV_FUNCTION_PRIVILEGES pour afficher les autorisations de fonction explicitement accordées aux utilisateurs, aux rôles et aux groupes de la base de données actuelle.

SVV_FUNCTION_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les identités auxquelles ils ont accès ou qu'ils possèdent.

Colonnes de la table

Nom de la colonne	Type de données	Description
namespace_name	text	Nom de l'espace de noms où existe une fonction spécifiée.

Nom de la colonne	Type de données	Description
function_name	text	Nom de la fonction.
argument_types	text	Chaîne qui représente le type de l'argument d'entrée d'une fonction.
privilege_type	text	Type de l'autorisation. La valeur possible est EXECUTE.
identity_id	entier	ID de l'identité. Les valeurs possibles sont ID d'utilisateur, ID de rôle ou ID de groupe.
identity_name	text	Nom de l'identité.
identity_type	text	Type d'identité. Les valeurs possibles sont utilisateur, rôle, groupe ou public.
admin_option	boolean	Valeur qui indique si l'utilisateur peut accorder l'autorisation à d'autres utilisateurs et rôles. Elle renvoie toujours une réponse fausse pour les types d'identités rôle et groupe.

Exemple de requête

L'exemple suivant montre le résultat de SVV_FUNCTION_PRIVILEGES.

```
SELECT
 namespace_name,function_name,argument_types,privilege_type,identity_name,identity_type,admin_o
 FROM svv_function_privileges
WHERE identity_name IN ('role1', 'reguser');

 namespace_name | function_name |       argument_types       | privilege_type |
 identity_name | identity_type | admin_option
----------------+---------------+----------------------------+----------------
+----------------+---------------+-------------
    public      | test_func1    | integer                    |    EXECUTE     |
 role1     |     role      |  False
```

```
    public      | test_func2    | integer, character varying |    EXECUTE     |
 reguser    |     user      |  False
```

SVV_GEOGRAPHY_COLUMNS

Utilisez SVV_GEOGRAPHY_COLUMNS pour afficher la liste des colonnes GEOGRAPHY dans votre entrepôt des données. Cette liste de colonnes inclut des colonnes provenant d'unités de partage des données.

SVV_GEOGRAPHY_COLUMNS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
f_table_catalog	varchar(128)	Nom de la base de données dans laquelle la table contenant la colonne GEOGRAPHY existe.
f_table_schema	varchar(128)	Nom du schéma dans lequel la table contenant la colonne GEOGRAPHY existe.
f_table_name	varchar(128)	Nom de la table dans laquelle la colonne GEOGRAPHY existe.
f_geography_column	varchar(128)	Nom de la colonne GEOGRAPHY.
coord_dimension	entier	Nombre de dimensions des données GEOGRAPHY.
srid	entier	Identifiant de système de référence spatiale (SRID) des données GEOGRAPHY.
type	varchar(128)	Nom du type de données de géographie spatiale.

Exemple de requête

L'exemple suivant montre le résultat de SVV_GEOGRAPHY_COLUMNS.

```
SELECT * FROM svv_geography_columns;

f_table_catalog  | f_table_schema  | f_table_name  | f_geography_column  |
 coord_dimension | srid |  type
-----------------+-----------------+--------------+--------------------
+-----------------+------+-------------
dev              | public          | spatial_test  | test_geography      | 2
    | 0    | GEOGRAPHY
```

SVV_GEOMETRY_COLUMNS

Utilisez SVV_GEOMETRY_COLUMNS pour afficher la liste des colonnes GEOMETRY dans votre entrepôt des données. Cette liste de colonnes inclut des colonnes provenant d'unités de partage des données.

SVV_GEOMETRY_COLUMNS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
f_table_catalog	varchar(128)	Nom de la base de données dans laquelle la table contenant la colonne GEOMETRY existe.
f_table_schema	varchar(128)	Nom du schéma dans lequel la table contenant la colonne GEOMETRY existe.
f_table_name	varchar(128)	Nom de la table dans laquelle la colonne GEOMETRY existe.
f_geometry_column	varchar(128)	Nom de la colonne GEOMETRY.

Nom de la colonne	Type de données	Description
coord_dimension	entier	Nombre de dimensions des données GEOMETRY.
srid	entier	Identifiant de système de référence spatiale (SRID) des données GEOMETRY.
type	varchar(128)	Nom du type de géométrie spatiale.

Exemple de requête

L'exemple suivant montre le résultat de SVV_GEOMETRY_COLUMNS.

```
SELECT * FROM svv_geometry_columns;

f_table_catalog  | f_table_schema  | f_table_name  | f_geometry_column   |
 coord_dimension | srid |  type
-----------------+-----------------+---------------+--------------------
+-----------------+------+--------------
dev              | public          | accomodations | shape               | 2
    | 0    | GEOMETRY
dev              | public          | zipcode       | wkb_geometry        | 2
    | 0    | GEOMETRY
```

SVV_IAM_PRIVILEGES

Utilisez SVV_IAM_PRIVILEGES pour afficher les privilèges IAM explicitement accordés aux utilisateurs, aux rôles et aux groupes.

SVV_IAM_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les entrées auxquelles ils ont accès.

Colonnes de la table

Nom de la colonne	Type de données	Description
iam_arn	text	Nom de l'espace de noms.
command_type	text	Types de privilèges. Les valeurs possibles sont COPY, UNLOAD, CREATE MODEL ou EXTERNAL FUNCTION.
identity_id	entier	ID d'identité. Les valeurs possibles sont ID d'utilisateur, ID de rôle ou ID de groupe.
identity_name	text	Nom d'identité.
identity_type	text	Type d'identité. Les valeurs possibles sont utilisateur, rôle, groupe ou public.

Exemples de requêtes

L'exemple suivant montre les résultats de SVV_IAM_PRIVILEGES.

```
SELECT * from SVV_IAM_PRIVILEGES ORDER BY IDENTITY_ID;
       iam_arn        | command_type | identity_id | identity_name | identity_type
----------------------+--------------+-------------+---------------+---------------
 default-aws-iam-role | COPY         |           0 | public        | public
 default-aws-iam-role | UNLOAD       |           0 | public        | public
 default-aws-iam-role | CREATE MODEL |           0 | public        | public
 default-aws-iam-role | EXFUNC       |           0 | public        | public
 default-aws-iam-role | COPY         |         106 | u1            | user
 default-aws-iam-role | UNLOAD       |         106 | u1            | user
 default-aws-iam-role | CREATE MODEL |         106 | u1            | user
 default-aws-iam-role | EXFUNC       |         106 | u1            | user
 default-aws-iam-role | COPY         |      118413 | r1            | role
 default-aws-iam-role | UNLOAD       |      118413 | r1            | role
 default-aws-iam-role | CREATE MODEL |      118413 | r1            | role
 default-aws-iam-role | EXFUNC       |      118413 | r1            | role
```

```
(12 rows)
```

SVV_IDENTITY_PROVIDERS

La vue SVV_IDENTITY_PROVIDERS renvoie le nom et les propriétés supplémentaires des fournisseurs d'identité. Pour plus d'informations sur la création d'un fournisseur d'identité, consultez CREATE IDENTITY PROVIDER.

SVV_IDENTITY_PROVIDERS n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
uid	entier	ID unique du fournisseur d'identité enregistré.
name	text	Nom du fournisseur d'identité.
type	text	Type de fournisseur d'identité.
instanceid	text	Différenciateur unique entre des instances du même type.
namespc	text	Préfixe d'espace de noms du fournisseur d'identité.
params	text	Objet JSON avec des paramètres pour le fournisseur d'identité.
activé	bool	Indique si le fournisseur d'identité est activé.

Exemples de requêtes

Pour afficher les propriétés d'un fournisseurs d'identité, exécutez une requête comme la suivante, après avoir créé des fournisseurs d'identité.

```
SELECT name, type, instanceid, namespc, params, enabled
FROM svv_identity_providers
ORDER BY 1;
```

L’exemple de sortie inclut des descriptions de paramètres.

```
       name       | type  |              instanceid              | namespc |

                                                 params

                       | enabled
------------------+-------+--------------------------------------+---------
+-------------------------------------------------------------------------------------------------------------------------------
+---------
 rs5517_azure_idp | azure | e40d4bb2-7670-44ae-bfb8-5db013221d73 | abc     |
 {"issuer":"https://login.microsoftonline.com/e40d4bb2-7670-44ae-bfb8-5db013221d73/
v2.0", "client_id":"871c010f-5e61-4fb1-83ac-98610a7e9110", "client_secret":,
 "audience":["https://analysis.windows.net/powerbi/connector/AmazonRedshift", "https://
analysis.windows.net/powerbi/connector/AWSRDS"]} | t
(1 row)
```

SVV_INTEGRATION

SVV_INTEGRATION affiche des détails sur la configuration des intégrations.

SVV_INTEGRATION n’est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Pour plus d'informations sur les intégrations sans ETL, consultez la section Intégrations sans ETL.

Colonnes de la table

Nom de la colonne	Type de données	Description
integration_id	character (128)	Identifiant associé à l’intégration.
target_database	character (128)	Base de données d’Amazon Redshift qui reçoit les données d’intégration.

Nom de la colonne	Type de données	Description
source	character (128)	Données source de l'intégration. Les types possibles incluent `MySQL` `PostgreSQL` , et `S3_EVENT_NOTIFICATIONS` .
state	character (128)	État de l'intégration. Les valeurs possibles incluent `PendingDbConnectState` , `SchemaDiscoveryState` , `CdcRefreshState` et `ErrorState` .
current_lag	bigint	Temps de latence actuel (en millisecondes) entre la source et la destination de l'intégration.
last_replicated_checkpoint	character (128)	Dernier point de contrôle répliqué.
total_tables_replicated	entier	Nombre total de tables actuellement à l'état répliqué.
total_tables_failed	entier	Nombre total de tables actuellement à l'état d'échec.
creation_time	timestamp	Heure (UTC) à laquelle l'intégration est créée. Il est défini comme l'heure à laquelle la base de données cible est créée à partir de l'intégration.
intervalle de rafraîchissement	entier	Intervalle de temps approximatif, en secondes, pour actualiser les données de la source zéro ETL vers la base de données cible.
source_database	character (128)	Nom de la base de données source.
is_history_mode	boolean	Une `TRUE` valeur indique que le mode historique est activé. A `FALSE` indique que le mode historique est désactivé.

Exemples de requêtes

La commande SQL suivante affiche les intégrations actuellement définies.

```
select * from svv_integration;

           integration_id              | target_database | source |      state
   | current_lag |      last_replicated_checkpoint     | total_tables_replicated |
 total_tables_failed |       creation_time       |  refresh_interval  | source_database
 | is_history_mode
--------------------------------------+-----------------+--------+----------------
+-------------+--------------------------------------+-------------------------
+---------------------+---------------------------+--------------------
+-----------------+-----------------
  99108e72-1cfd-414f-8cc0-0216acefac77 |     perfdb      |  MySQL | CdcRefreshState |
  56606106  | {"txn_seq":9834,"txn_id":126597515} |           152          |
 0         | 2023-09-19 21:05:27.520299|      720           + mysourceetl     | f
```

SVV_INTEGRATION_TABLE_MAPPAGE

SVV_INTEGRATION_TABLE_MAPPING affiche le mappage de la base de données source, du schéma, de la table, de la colonne et du type de données vers la cible lorsque la valeur d'identifiant de ces champs est différente.

> **Note**
>
> Cette vue est uniquement renseignée pour les types d'intégrations sans ETL suivants :
>
> - AWS Glue applications tierces pour Amazon SageMaker Lakehouse
> - Amazon DynamoDB vers Amazon Lakehouse SageMaker
>
> Pour plus d'informations, consultez la section Intégrations Zero-ETL dans le Guide du AWS Glue développeur.

La transformation des valeurs d'identifiant de la source vers la cible suit les règles suivantes :

- Une majuscule est convertie en minuscule.
- Un caractère autre qu'une lettre minuscule, un chiffre ou un trait de soulignement (_) est converti en trait de soulignement (_).

- En cas de conflit avec une valeur d'identifiant existante, un identifiant unique universel (UUID) est ajouté au nouvel identifiant.
- Si la valeur de l'identifiant source est un mot clé Amazon Redshift, le suffixe `_redshift` est ajouté au nouvel identifiant.

Après la transformation, un caractère doit être une lettre minuscule, un chiffre ou un trait de soulignement (_) et correspondre au modèle regex. `[a-z0-9_]` Les exemples suivants illustrent les règles de conversion :

Source	Cible	Remarques
foo	foo	Aucune transformation
Bar (À barres)	bar	
FooBar	foobar	
foo1	foo1	Aucune transformation
foo_1	foo_1	Aucune transformation
Bar @1	bar_1	
foo_bar@	foo_bar_	
cas	cas_redshift	

SVV_INTEGRATION_TABLE_MAPPING est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Pour plus d'informations sur les intégrations zéro ETL, consultez la section Intégrations zéro ETL dans le guide de gestion Amazon Redshift.

Colonnes de la table

Nom de la colonne	Type de données	Description
integration_id	character (128)	Identifiant associé à l’intégration.
source_database	character (128)	Nom de la base de données dans la source.
target_database	character (128)	Base de données d’Amazon Redshift qui reçoit les données d’intégration.
nom_schéma_source	character (128)	Le nom du schéma dans la source.
nom_schéma_cible	character (128)	Schéma dans Amazon Redshift qui reçoit les données d'intégration.
nom_table_source	character (128)	Nom de la table dans la source.
nom_table_cible	character (128)	La table d'Amazon Redshift qui reçoit les données d'intégration.
nom_chronique_source	character (128)	Le nom de la colonne dans la source.
nom_colonne cible	character (128)	Colonne d'Amazon Redshift qui reçoit les données d'intégration.
type_données_source	character (128)	Type de données de la colonne dans la source.
type de données_cible	character (128)	Type de données de la colonne dans Amazon Redshift qui reçoit les données d'intégration.

Exemples de requêtes

La commande SQL suivante affiche le mappage des valeurs de métadonnées de la source à la cible.

```
select * from svv_integration_table_mapping;

           integration_id              | source_database | target_database |
 source_schema_name | target_schema_name | source_table_name | target_table_name |
--------------------------------------+-----------------+-----------------
+--------------------+--------------------+-------------------------------------+
  99108e72-1cfd-414f-8cc0-0216acefac77 |     mydatabase  |  mydatabase     |  myschema
         | myschema           | Mytable           | mytable           |


                                       | source_column_name | target_column_name |
 source_data_type | target_data_type |
                                      +--------------------+--------------------
+------------------+------------------+
                                       | Mycolumnname       | mycolumnname       |
 Mydatatype       | mydatatype       |
```

SVV_INTEGRATION_TABLE_STATE

SVV_INTEGRATION_TABLE_STATE affiche des détails sur les informations d'intégration au niveau de la table.

SVV_INTEGRATION_TABLE_STATE n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Pour plus d'informations, consultez la section Intégrations Zero-ETL.

Colonnes de la table

Nom de la colonne	Type de données	Description
integration_id	character(128)	Identifiant associé à l'intégration.
target_database	character(128)	Nom de la base de données Amazon Redshift.
nom_schéma	character(128)	Nom du schéma Amazon Redshift.
table_name	character(128)	Nom de la table.

Nom de la colonne	Type de données	Description
table_state	character(128)	État de la table. Les valeurs possibles sont `SyncedFailed`,`Deleted`,`ResyncRequired` , `ResyncInitiated` et`DroppedSource` . L'`DroppedSource` état indique que la source d'une table en mode historique a été supprimée à la source.
table_last_replicated_checkpoint	character(128)	Coordonnées du journal synchronisé actuel.
raison	character(256)	Motif de la dernière transition d'état. Les raisons les plus courantes sont : types de données non pris en charge dans les tables ou absence de clés primaires dans les tables. Pour en savoir plus sur la résolution des problèmes courants, consultez Résolution des problèmes liés aux intégrations zéro ETL dans Amazon Redshift.
last_updated_timestamp	horodatage sans fuseau horaire	Heure (UTC) de la dernière mise à jour de la table.
table_lignes	bigint	Nombre total de lignes de la table.
table_size	bigint	Taille de la table en mégaoctets (Mo).
is_history_mode	boolean	Une `TRUE` valeur indique que le mode historique est activé. A `FALSE` indique que le mode historique est désactivé.

Exemples de requêtes

La commande SQL suivante affiche les colonnes du journal des intégrations.

```
select * from svv_integration_table_state;
```

```
          integration_id              | target_database | schema_name |     table_name
   | table_state  |table_last_replicated_checkpoint | reason | last_updated_timestamp
   |table_rows  | table_size | is_history_mode
-------------------------------------+-----------------+------------
+------------------+-------------+---------------------------------+--------
+---------------------------+-----------+-----------+----------------
 4798e675-8f9f-4686-b05f-92c538e19629 |  sample_test2   |    sample   |
 SampleTestChannel |    Synced    |   {"txn_seq":3,"txn_id":3122}   |        |
 2023-05-12 12:40:30.656625 | 2          |   16       | f
```

SVV_INTERLEAVED_COLUMNS

Utilisez la vue SVV_INTERLEAVED_COLUMNS pour aider à déterminer si une table qui utilise les clés de tri entrelacé doit être réindexée à l'aide de VACUUM REINDEX. Pour plus d'informations sur la façon de déterminer la fréquence d'exécution de VACUUM et sur le moment d'exécution d'un VACUUM REINDEX, consultez Minimiser les temps d'aspiration.

SVV_INTERLEAVED_COLUMNS n'est visible que par les super-utilisateurs. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
tbl	entier	ID de table.
col	entier	Index de base zéro de la colonne.
interleaved_skew	numeric 9,2)	Rapport qui indique le delta présent dans les colonnes de clés de tri entrelacé d'une table. Une valeur égale à 1,00 indique l'absence de delta et les valeurs plus élevées un delta plus important. Les tables avec un delta important doivent être réindexées avec la commande VACUUM REINDEX.
last_reindex	timestar	Heure à laquelle le dernier VACUUM REINDEX a été exécuté pour la table spécifiée. Cette valeur est NULL si une table n'a jamais été réindexée ou si la table de journal système sous-jacente, STL_VACUUM, a fait l'objet d'une rotation depuis la dernière réindexation.

Exemples de requêtes

Pour identifier les tables susceptibles d'être réindexées, exécutez la requête suivante.

```
select tbl as tbl_id, stv_tbl_perm.name as table_name,
col, interleaved_skew, last_reindex
from svv_interleaved_columns, stv_tbl_perm
where svv_interleaved_columns.tbl = stv_tbl_perm.id
and interleaved_skew is not null;

 tbl_id | table_name | col | interleaved_skew | last_reindex
--------+------------+-----+------------------+---------------------
 100068 | lineorder  |   0 |             3.65 | 2015-04-22 22:05:45
 100068 | lineorder  |   1 |             2.65 | 2015-04-22 22:05:45
 100072 | customer   |   0 |             1.65 | 2015-04-22 22:05:45
 100072 | lineorder  |   1 |             1.00 | 2015-04-22 22:05:45
(4 rows)
```

SVV_LANGUAGE_PRIVILEGES

Utilisez SVV_LANGUAGE_PRIVILEGES pour afficher les autorisations de langage explicitement accordées aux utilisateurs, aux rôles et aux groupes de la base de données actuelle.

SVV_LANGUAGE_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les identités auxquelles ils ont accès ou qu'ils possèdent.

Colonnes de la table

Nom de la colonne	Type de données	Description
language_ name	text	Nom du langage.

Nom de la colonne	Type de données	Description
privilege_type	text	Type de l'autorisation. La valeur possible est USAGE.
identity_id	entier	ID de l'identité. Les valeurs possibles sont ID d'utilisateur, ID de rôle ou ID de groupe.
identity_name	text	Nom de l'identité.
identity_type	text	Type d'identité. Les valeurs possibles sont utilisateur, rôle, groupe ou public.
admin_option	boolean	Valeur qui indique si l'utilisateur peut accorder l'autorisation à d'autres utilisateurs et rôles. Elle renvoie toujours une réponse fausse pour les types d'identités rôle et groupe.

Exemple de requête

L'exemple suivant montre le résultat de SVV_LANGUAGE_PRIVILEGES.

```
SELECT language_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_language_privileges
WHERE identity_name IN ('role1', 'reguser');

 language_name | privilege_type | identity_name | identity_type | admin_option
---------------+----------------+---------------+---------------+--------------
   exfunc      |     USAGE      |    reguser    |     user      |    False
   exfunc      |     USAGE      |     role1     |     role      |    False
   plpythonu   |     USAGE      |    reguser    |     user      |    False
```

SVV_MASKING_POLICY

Utilisez SVV_MASKING_POLICY pour afficher toutes les politiques de masquage créées sur le cluster.

Seuls les super-utilisateurs ayant le rôle `sys:secadmin` peuvent afficher SVV_MASKING_POLICY. Les utilisateurs réguliers verront 0 ligne.

Colonnes de la table

Nom de la colonne	Type de données	Description
policy_database	text	Nom de la base de données dans laquelle la politique de masquage a été créée.
policy_name	text	Nom de la politique de masquage.
input_columns	text	Attributs fournis dans la clause WITH de l'instruction CREATE POLICY.
policy_expression	text	Expression de masquage utilisée dans la politique.
policy_modified_by	text	Nom de l'utilisateur qui a modifié la politique pour la dernière fois.
policy_modified_time	timestamp	Horodatage de la création ou de la dernière modification de la politique.

SVV_ML_MODEL_INFO

Informations sur l'état actuel du modèle de machine learning.

SVV_ML_MODEL_INFO est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	char(128)	Base de données du modèle.
nom_schéma	char(128)	Schéma du modèle.
user_name	char(128)	Le propriétaire du modèle.
model_name	char(128)	Nom du modèle.
life_cycle	char(20)	L'état du cycle de vie du modèle.
is_refreshable	entier	L'état du modèle, à savoir s'il peut être actualisé si les tables et les colonnes originales de la requête d'entraînement existent toujours et si l'utilisateur a toujours les permissions pour les utiliser. Les valeurs possibles sont : 1 (actualis able) et 0 (non actualisable).
model_state	char(128)	L'état actuel du modèle.

Exemple de requête

La requête suivante affiche l'état actuel des modèles de machine learning.

```
SELECT schema_name, model_name, model_state
FROM svv_ml_model_info;

 schema_name |        model_name             |             model_state
-------------+-------------------------------+-------------------------------------
 public      | customer_churn_auto_model     | Train Model On SageMaker In Progress
 public      | customer_churn_xgboost_model | Model is Ready
(2 row)
```

SVV_ML_MODEL_PRIVILEGES

Utilisez SVV_ML_MODEL_PRIVILEGES pour afficher les autorisations de modèle de machine learning explicitement accordées aux utilisateurs, aux rôles et aux groupes du cluster.

SVV_ML_MODEL_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les identités auxquelles ils ont accès ou qu'ils possèdent.

Colonnes de la table

Nom de la colonne	Type de données	Description
namespace_name	text	Nom de l'espace de noms où existe un modèle de machine learning spécifié.
model_name	text	Nom du modèle de machine learning.
modèle_version	entier	Numéro de version du modèle.
privilege_type	text	Type de l'autorisation. La valeur possible est EXECUTE.
identity_id	entier	ID de l'identité. Les valeurs possibles sont ID d'utilisateur, ID de rôle ou ID de groupe.
identity_name	text	Nom de l'identité.
identity_type	text	Type d'identité. Les valeurs possibles sont utilisateur, rôle, groupe ou public.

Nom de la colonne	Type de données	Description
admin_option	boolean	Valeur qui indique si l'utilisateur peut accorder l'autorisation à d'autres utilisateurs et rôles. Elle renvoie toujours une réponse fausse pour les types d'identités rôle et groupe.

Exemple de requête

L'exemple suivant montre le résultat de SVV_ML_MODEL_PRIVILEGES.

```
SELECT
 namespace_name,model_name,model_version,privilege_type,identity_name,identity_type,admin_optio
 FROM svv_ml_model_privileges
WHERE model_name = 'test_model';

 namespace_name | model_name | model_version | privilege_type |  identity_name |
 identity_type | admin_option
----------------+------------+---------------+----------------+----------------
+---------------+--------------
      public    | test_model |       1       |    EXECUTE     |     reguser    |
 user       |    False
      public    | test_model |       1       |    EXECUTE     |     role1      |
 role       |    False
```

SVV_MV_DEPENDENCY

La table SVV_MV_DEPENDENCY affiche les dépendances des vues matérialisées sur d'autres vues matérialisées dans Amazon Redshift.

Pour plus d'informations sur les vues matérialisées, consultez Vues matérialisées dans Amazon Redshift.

SVV_MV_DEPENDENCY est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	char(128)	La base de données qui contient la vue matérialisée spécifiée.
nom_schéma	char(128)	Schéma de la vue matérialisée.
name	char(128)	Nom de la vue matérialisée.
dependent_database_name	char(128)	Base de données de vue matérialisée dont dépend cette vue matérialisée.
dependent_schema_name	char(128)	Le schéma de vue matérialisée dont dépend cette vue matérialisée.
dependent_name	char(128)	Le nom de la vue matérialisée dont dépend cette vue matérialisée.

Exemple de requête

La requête suivante renvoie une ligne de sortie qui indique que la vue matérialisée `mv_over_foo` utilise la vue matérialisée `mv_foo` dans sa définition en tant que dépendance.

```
CREATE SCHEMA test_ivm_setup;
CREATE TABLE test_ivm_setup.foo(a INT);
CREATE MATERIALIZED VIEW test_ivm_setup.mv_foo AS SELECT * FROM test_ivm_setup.foo;
CREATE MATERIALIZED VIEW test_ivm_setup.mv_over_foo AS SELECT * FROM
 test_ivm_setup.mv_foo;

SELECT * FROM svv_mv_dependency;

 database_name | schema_name          | name        | dependent_database_name |
 dependent_schema_name     | dependent_name
---------------+----------------------+-------------+-------------------------
+---------------------------+----------
```

```
dev           | test_ivm_setup       | mv_over_foo |                    dev |
test_ivm_setup            | mv_foo
```

SVV_MV_INFO

La table SVV_MV_INFO contient une ligne pour chaque vue matérialisée, si les données sont obsolètes, ainsi que des informations d'état.

Pour plus d'informations sur les vues matérialisées, consultez Vues matérialisées dans Amazon Redshift.

SVV_MV_INFO est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	char(128)	Base de données contenant la vue matérialisée.
nom_schéma	char(128)	Schéma de la base de données.
user_name	char(128)	Utilisateur propriétaire de la vue matérialisée.
name	char(128)	Nom de la vue matérialisée.
is_stale	char(1)	`t` indique que la vue matérialisée est obsolète. Une vue matérialisée obsolète est une vue dans laquelle les tables de base ont été mises à jour mais où la vue matérialisée n'a pas été actualisé e. Ces informations peuvent ne pas être exactes si une actualisation n'a pas été exécutée depuis le dernier redémarrage.
state	entier	État de la vue matérialisée comme suit : • 0 – La vue matérialisée est entièrement recalculée lors de l'actualisation.

Nom de la colonne	Type de données	Description
		• 1 – La vue matérialisée est incrémentielle. • 101 – La vue matérialisée ne peut pas être actualisée en raison d'une colonne supprimée. Cette contrainte s'applique même si la colonne n'est pas utilisée dans la vue matérialisée. • 102 – La vue matérialisée ne peut pas être actualisée en raison d'un type de colonne modifié. Cette contrainte s'applique même si la colonne n'est pas utilisée dans la vue matérialisée. • 103 – La vue matérialisée ne peut pas être actualisée en raison d'une table renommée. • 104 – La vue matérialisée ne peut pas être actualisée en raison d'une colonne renommée. Cette contrainte s'applique même si la colonne n'est pas utilisée dans la vue matérialisée. • 105 – La vue matérialisée ne peut pas être actualisée en raison d'un schéma renommé.
réécriture automatique	char(1)	Un `t` indique que la vue matérialisée est éligible pour la réécriture automatique des requêtes.
actualisation automatique	char(1)	Un `t` indique que la vue matérialisée peut être automatiquement actualisée.

Exemple de requête

Pour afficher l'état de toutes les vues matérialisées, exécutez la requête suivante.

```
select * from svv_mv_info;
```

Cette requête renvoie l'exemple de sortie suivant.

```
database_name |       schema_name       | user_name |   name   |  is_stale | state |
 autorefresh | autorewrite
-------------+-------------------------+-----------+---------+-----------+-------
+------------+---------------
 dev          | test_ivm_setup          | catch-22  | mv       |   f       |     1 |
       1 |           0
 dev          | test_ivm_setup          | lotr      | old_mv   |   t       |     1 |
       0 |           1
```

SVV_QUERY_INFLIGHT

Utilisez la vue SVV_INFLIGHT pour déterminer quelles requêtes sont en cours d'exécution sur la base de données. Cette vue joint STV_INFLIGHT à STL_QUERYTEXT. SVV_QUERY_INFLIGHT n'affiche pas les requêtes de nœud principal uniquement. Pour plus d'informations, consultez Fonctions exécutées uniquement sur le nœud principal.

SVV_QUERY_INFLIGHT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> Cette vue n'est disponible que lors de l'interrogation des clusters alloués.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
slice	entier	Tranche sur laquelle la requête s'exécute.
query	entier	ID de requête. Permet de joindre d'autres tables système et vues.

Nom de la colonne	Type de données	Description
pid	entier	ID du processus. Toutes les requêtes d'une séance étant exécutées dans le même processus, cette valeur reste constante si vous exécutez une série de requêtes dans la même séance. Vous pouvez utiliser cette colonne pour la joindre à la table STL_ERROR.
starttime	timestamp	Heure de début de la requête.
suspended	entier	Indique si la requête est suspendue : **0** = false; **1** = true.
text	character(200)	Texte de la requête, par incréments de 200 caractères.
sequence	entier	Numéro de séquence pour les segments des instructions de requête.

Exemples de requêtes

L'exemple de sortie ci-dessous affiche deux requêtes en cours d'exécution, la requête SVV_QUERY_INFLIGHT elle-même et la requête 428, qui est divisée en trois lignes de la table. (Les colonnes starttime et statement sont tronquées dans cet exemple de sortie.)

```
select slice, query, pid, starttime, suspended, trim(text) as statement, sequence
from svv_query_inflight
order by query, sequence;

slice|query| pid  |      starttime       |suspended| statement | sequence
-----+-----+------+----------------------+---------+-----------+---------
1012 | 428 | 1658 | 2012-04-10 13:53:... |       0 | select ...|    0
1012 | 428 | 1658 | 2012-04-10 13:53:... |       0 | enueid ...|    1
1012 | 428 | 1658 | 2012-04-10 13:53:... |       0 | atname,...|    2
1012 | 429 | 1608 | 2012-04-10 13:53:... |       0 | select ...|    0
(4 rows)
```

SVV_QUERY_STATE

Utilisez SVV_QUERY_STATE pour afficher les informations sur l'exécution des requêtes en cours.

La vue SVV_QUERY_STATE contient un sous-ensemble de données de la table STV_EXEC_STATE.

SVV_QUERY_STATE est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

> **Note**
>
> Cette vue n'est disponible que lors de l'interrogation des clusters alloués.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. Permet de joindre d'autres tables système et vues.
seg	entier	Numéro du segment de requête en cours d'exécution. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Les segments de requête peuvent s'exécuter en parallèle. Chaque segment s'exécute dans un processus unique.
étape	entier	Numéro de l'étape de requête en cours d'exécution. Une étape est la plus petite unité de l'exécution des requêtes. Chaque étape représente une unité de travail discrète, telle que l'analyse d'une table, le retour de résultats ou le tri de données.

Nom de la colonne	Type de données	Description
maxtime	interval	Durée maximale (en microsecondes) d'exécution de l'étape.
avgtime	interval	Durée moyenne (en microsecondes) d'exécution de l'étape.
rows	bigint	Nombre de lignes générées par l'étape en cours d'exécution.
bytes	bigint	Nombre d'octets générés par l'étape en cours d'exécution.
cpu	bigint	Pour utilisation interne.
memory	bigint	Pour utilisation interne.
rate_row	double precision	Rows-per-second taux depuis le début de la requête, calculé en additionnant les lignes et en divisant par le nombre de secondes entre le début de la requête et l'heure actuelle.
rate_byte	double precision	Bytes-per-second taux depuis le début de la requête, calculé en additionnant les octets et en divisant par le nombre de secondes entre le début de la requête et l'heure actuelle.
étiquette	character(25)	Étiquette de requête : nom pour l'étape, comme `scan` ou `sort`.
is_diskbased	character(1)	Indique si l'étape de la requête s'exécute comme opération sur disque : true (**`t`**) ou false (**`f`**). Seules certaines étapes, telles que le hachage, le tri et l'agrégation, peuvent accéder au disque. La plupart des types d'étapes sont toujours exécutés en mémoire.
workmem	bigint	Quantité de mémoire de travail (en octets) attribuée à l'étape de requête.

Nom de la colonne	Type de données	Description
num_parts	entier	Nombre de partitions entre lesquelles une table de hachage est divisée pendant une étape de hachage. Un nombre positif dans cette colonne n'implique pas que l'étape de hachage ait été exécutée comme opération sur disque. Vérifiez la valeur de la colonne IS_DISKBASED pour voir si l'étape de hachage était basée sur le disque.
is_rrscan	character(1)	Si la valeur est définie sur true (**t**), indique qu'une analyse à plage restreinte a été utilisée sur l'étape. La valeur par défaut est false (**f**).
is_delayed_scan	character(1)	Si la valeur est définie sur true (**t**), indique qu'une analyse retardée a été utilisée sur l'étape. La valeur par défaut est false (**f**).

Exemples de requêtes

Détermination du temps de traitement d'une requête par étape

La requête suivante affiche le délai d'exécution de chaque étape de la requête avec l'ID de requête 279 et le nombre de lignes de données traitées par Amazon Redshift :

```
select query, seg, step, maxtime, avgtime, rows, label
from svv_query_state
where query = 279
order by query, seg, step;
```

Cette requête extrait les informations de traitement de la requête 279, comme illustré dans l'exemple de sortie suivant :

```
query |   seg   | step | maxtime | avgtime |  rows   | label
------+---------+------+---------+---------+---------+-------------------
  279 |       3 |    0 | 1658054 | 1645711 | 1405360 | scan
  279 |       3 |    1 | 1658072 | 1645809 |       0 | project
  279 |       3 |    2 | 1658074 | 1645812 | 1405434 | insert
  279 |       3 |    3 | 1658080 | 1645816 | 1405437 | distribute
```

```
  279 |       4 |    0 | 1677443 | 1666189 | 1268431 | scan
  279 |       4 |    1 | 1677446 | 1666192 | 1268434 | insert
  279 |       4 |    2 | 1677451 | 1666195 |       0 | aggr
(7 rows)
```

Détermination si des requêtes actives sont en cours d'exécution sur le disque

La requête suivante affiche si des requêtes actives sont en cours d'exécution sur le disque :

```
select query, label, is_diskbased from svv_query_state
where is_diskbased = 't';
```

Cet exemple de sortie affiche les requêtes actives en cours d'exécution sur le disque :

```
 query | label        | is_diskbased
-------+--------------+--------------
1025   | hash tbl=142 |      t
(1 row)
```

SVV_REDSHIFT_COLUMNS

Utilisez SVV_REDSHIFT_COLUMNS pour afficher la liste de toutes les colonnes auxquelles un utilisateur a accès. Cet ensemble de colonnes comprend les colonnes du cluster et les colonnes des datashares fournis par les clusters distants.

SVV_REDSHIFT_COLUMNS est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	varchar(128)	Le nom de la base de données où figure la table contenant les colonnes.
nom_schéma	varchar(128)	Nom du schéma de la table.
table_name	varchar(128)	Nom de la table.

Nom de la colonne	Type de données	Description
column_name	varchar(128)	Nom de la colonne.
ordinal_position	entier	Position de la colonne dans la table.
data_type	varchar(32)	Type de données de la colonne.
column_default	varchar(4000)	Valeur par défaut de la colonne.
is_nullable	varchar(3)	Valeur qui définit si la colonne est nullable. Les valeurs possibles sont `yes`, `no` et " " (chaîne vide) qui ne représente aucune informati on.
encoding	varchar(128)	Type d'encodage de la colonne.
distkey	boolean	Une valeur définie sur TRUE si cette colonne est la clé de distribution de la table. Sinon, elle est définie sur FALSE.

Nom de la colonne	Type de données	Description
sortkey	entier	Une valeur qui spécifie l'ordre de la colonne dans la clé de tri. Si la table utilise une clé de tri composée et que toutes les colonnes qui font partie de la clé de tri ont une valeur positive qui indique la position de la colonne dans la clé de tri. Si la table utilise une clé de tri entrelacée, chaque colonne qui fait partie de la clé de tri a une valeur qui est tour à tour positive ou négative. Ici, la valeur absolue indique la position de la colonne dans la clé de tri. Si `sortkey` est 0, la colonne ne fait pas partie d'une clé de tri.
column_acl	varchar(128)	Chaîne qui définit les autorisations de l'utilisateur ou du groupe d'utilisateurs spécifié pour la colonne.
remarks	varchar(256)	Remarques.

Exemple de requête

L'exemple suivant renvoie la sortie de SVV_REDSHIFT_COLUMNS.

```
SELECT *
FROM svv_redshift_columns
WHERE database_name = 'tickit_db'
    AND TABLE_NAME = 'tickit_sales_redshift'
ORDER BY COLUMN_NAME,
    TABLE_NAME,
    database_name
LIMIT 5;

database_name | schema_name |       table_name      | column_name | ordinal_position |
 data_type | column_default | is_nullable | encoding | distkey | sortkey | column_acl
 | remarks
--------------+-------------+-----------------------+-------------+------------------
+-----------+----------------+-------------+----------+---------+---------
+-------------+--------
   tickit_db  |   public    | tickit_sales_redshift |   buyerid   |        4         |
 integer  |                |      NO     |   az64   |  False  |    0    |             |
   tickit_db  |   public    | tickit_sales_redshift |  commission |        9         |
 numeric  |      (8,2)     |     YES     |   az64   |  False  |    0    |             |
   tickit_db  |   public    | tickit_sales_redshift |    dateid   |        6         |
 smallint |                |      NO     |   none   |  False  |    1    |             |
   tickit_db  |   public    | tickit_sales_redshift |   eventid   |        5         |
 integer  |                |      NO     |   az64   |  False  |    0    |       |
   tickit_db  |   public    | tickit_sales_redshift |   listid    |        2         |
 integer  |                |      NO     |   az64   |  True   |    0    |             |
```

SVV_REDSHIFT_DATABASES

Utilisez SVV_REDSHIFT_DATABASES pour afficher la liste de toutes les bases de données auxquelles un utilisateur a accès. Cela inclut les bases de données sur le cluster et les bases de données créées à partir des datashares fournis par les clusters distants.

SVV_REDSHIFT_DATABASES est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	varchar(128)	Nom de la base de données.

Nom de la colonne	Type de données	Description
database_owner	entier	L'ID utilisateur du propriétaire de la base de données.
database_type	varchar(32)	Type de base de données. Les types possibles sont les bases de données locales ou partagées.
database_acl	varchar(128)	Information à utilisation interne uniquement.
database_options	varchar(128)	Les propriétés de la base de données.
database_isolation_level	varchar(128)	Niveau d'isolation de la base de données. Les valeurs possibles incluent `Snapshot Isolation` et `Serializable`.

Exemple de requête

L'exemple suivant renvoie la sortie pour SVV_REDSHIFT_DATABASES.

```
select database_name, database_owner, database_type, database_options,
 database_isolation_level
from  svv_redshift_databases;


database_name | database_owner | database_type | database_options |
 database_isolation_level
--------------+----------------+---------------+------------------+------------------
   dev        |  1             | local         | NULL             | Serializable
```

SVV_REDSHIFT_FUNCTIONS

Utilisez SVV_REDSHIFT_FUNCTIONS pour afficher la liste de toutes les fonctions auxquelles un utilisateur a accès. Cet ensemble de fonctions comprend les fonctions du cluster et celles des datashares fournis par les clusters distants.

SVV_REDSHIFT_FUNCTIONS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	varchar(128)	Le nom de la base de données où figure le cluster qui dispose de ces fonctions.
nom_schéma	varchar(128)	Le nom du schéma qui spécifie une fonction donnée.
function_name	varchar(128)	Le nom d'une fonction spécifiée.
function_type	varchar(128)	Le type de fonction. Les valeurs possibles sont les fonctions normales, les fonctions d'agrégat et les procédures stockées.
argument_type	varchar(512)	Une chaîne qui représente le type de l'argument d'entrée d'une fonction.
result_type	varchar(128)	Le type de données de la valeur de retour d'une fonction.

Exemple de requête

L'exemple suivant renvoie la sortie de SVV_REDSHIFT_FUNCTIONS.

```
SELECT *
FROM svv_redshift_functions
WHERE database_name = 'tickit_db'
    AND SCHEMA_NAME = 'public'
ORDER BY function_name
LIMIT 5;

database_name | schema_name |      function_name    |  function_type   |
 argument_type  | result_type
--------------+-------------+-----------------------+------------------
+------------------+-------------
   tickit_db  |    public   |     shared_function   | REGULAR FUNCTION | integer,
 integer |   integer
```

SVV_REDSHIFT_SCHEMA_QUOTA

Affiche le quota et l'utilisation actuelle du disque pour chaque schéma dans une base de données.

SVV_REDSHIFT_SCHEMA_QUOTA est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Cette vue est disponible lors de l'interrogation de clusters provisionnés ou de groupes de travail Redshift sans serveur.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	character(128)	La base de données qui contient le schéma.
nom_schéma	character(128)	Nom du schéma.
schema_owner	entier	ID utilisateur interne du propriétaire du schéma.

Nom de la colonne	Type de données	Description
quota	entier	Quantité d'espace disque (en Mo) que le schéma peut utiliser.
disk_usage	entier	Espace disque (en Mo) actuellem ent utilisé par le schéma.

Exemple de requête

L'exemple suivant montre comment afficher le quota et l'utilisation actuelle du disque pour le schéma nommé `sales_schema`.

```
SELECT TRIM(SCHEMA_NAME) "schema_name", QUOTA, disk_usage FROM
 svv_redshift_schema_quota
WHERE SCHEMA_NAME = 'sales_schema';


schema_name   | quota | disk_usage
--------------+-------+------------
sales_schema  | 2048  | 30
```

SVV_REDSHIFT_SCHEMAS

Utilisez SVV_REDSHIFT_SCHEMAS pour afficher la liste de tous les schémas auxquels un utilisateur a accès. Cet ensemble de schémas comprend les schémas du cluster et ceux des unités de partage des données fournies par les clusters distants.

SVV_REDSHIFT_SCHEMAS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	varchar(128)	Le nom de la base de données où existe un schéma spécifié.
nom_schéma	varchar(128)	Nom de l'espace de noms ou du schéma.
schema_owner	entier	ID utilisateur interne du propriétaire du schéma.
schema_type	varchar(16)	Type de schéma. Les valeurs possibles sont les schémas partagés et locaux.
schema_acl	varchar(128)	Chaîne qui définit les autorisations de l'utilisateur ou du groupe d'utilisateurs spécifié pour le schéma.
schema_option	varchar(128)	Les options du schéma.

Exemple de requête

L'exemple suivant renvoie la sortie de SVV_REDSHIFT_SCHEMAS.

```
SELECT *
FROM svv_redshift_schemas
WHERE database_name = 'tickit_db'
ORDER BY database_name,
    SCHEMA_NAME;

database_name |    schema_name     | schema_owner | schema_type | schema_acl |
 schema_option
--------------+--------------------+--------------+-------------+------------
+---------------
```

```
 tickit_db  |      public      |      1      |    shared   |           |
```

SVV_REDSHIFT_TABLES

Utilisez SVV_REDSHIFT_TABLES pour afficher la liste de toutes les tables auxquelles un utilisateur a accès. Cet ensemble de tables comprend les tables du cluster et celles des datashares fournis par les clusters distants.

SVV_REDSHIFT_TABLES est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database_name	varchar(128)	Le nom de la base de données où existe une table spécifiée.
nom_schéma	varchar(128)	Nom du schéma de la table.
table_name	varchar(128)	Nom de la table.
table_type	varchar(128)	Type de la table. Les valeurs possibles sont les vues et les tables.
table_acl	varchar(128)	Chaîne qui définit les autorisations de l'utilisateur ou du groupe d'utilisateurs spécifié pour la table.
remarks	varchar(128)	Remarques.
table_owner	varchar(128)	Propriétaire de la table.

Exemple de requête

L'exemple suivant renvoie la sortie de SVV_REDSHIFT_TABLES.

```
SELECT *
FROM svv_redshift_tables
WHERE database_name = 'tickit_db' AND TABLE_NAME LIKE 'tickit_%'
ORDER BY database_name,
TABLE_NAME;

database_name | schema_name |         table_name       | table_type | table_acl |
 remarks | table_owner
--------------+-------------+--------------------------+------------+-----------
+---------+-----------
   tickit_db  |    public   | tickit_category_redshift |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_date_redshift   |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_event_redshift  |    TABLE   |           |
   +
   tickit_db  |    public   |  tickit_listing_redshift |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_sales_redshift  |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_users_redshift  |    TABLE   |           |
   +
   tickit_db  |    public   |   tickit_venue_redshift  |    TABLE   |           |
```

Si la valeur table_acl est nulle, aucun privilège d'accès n'a été explicitement accordé à la table correspondante.

SVV_RELATION_PRIVILEGES

Utilisez SVV_RELATION_PRIVILEGES pour afficher les autorisations de relation (tables et vues) explicitement accordées aux utilisateurs, aux rôles et aux groupes de la base de données actuelle.

SVV_RELATION_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation SYSLOG ACCESS UNRESTRICTED

Les autres utilisateurs ne peuvent consulter que les identités auxquelles ils ont accès ou qu'ils possèdent. Pour plus d'informations sur la visibilité des données, consultezVisibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
namespace _name	text	Nom de l'espace de noms où existe une relation spécifiée.
relation_name	text	Nom de la relation.
privilege_type	text	Type de l'autorisation. Les valeurs possibles sont INSERT, SELECT, UPDATE, DELETE, REFERENCES ou DROP.
identity_id	entier	ID de l'identité. Les valeurs possibles sont ID d'utilisateur, ID de rôle ou ID de groupe.
identity_name	text	Nom de l'identité.
identity_type	text	Type d'identité. Les valeurs possibles sont utilisateur, rôle, groupe ou public.
admin_option	boolean	Valeur qui indique si l'utilisateur peut accorder l'autorisation à d'autres utilisateurs et rôles. Elle renvoie toujours une réponse fausse pour les types d'identités rôle et groupe.

Exemple de requête

L'exemple suivant montre le résultat de SVV_RELATION_PRIVILEGES.

```
SELECT
 namespace_name,relation_name,privilege_type,identity_name,identity_type,admin_option
 FROM svv_relation_privileges
WHERE relation_name = 'orders' AND privilege_type = 'SELECT';

 namespace_name | relation_name | privilege_type |  identity_name | identity_type |
 admin_option
```

```
---------------+--------------+---------------+---------------+--------------
+-------------
     public     |    orders    |     SELECT    |    reguser    |     user     |
 False
     public     |    orders    |     SELECT    |     role1     |     role     |
 False
```

SVV_RLS_APPLIED_POLICY

Utilisez SVV_RLS_APPLIED_POLICY pour suivre l'application des politiques RLS sur les requêtes qui font référence à des relations protégées par la RLS.

SVV_RLS_APPLIED_POLICY est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant du rôle `sys:operator`
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Remarquez que sys:secadmin ne dispose pas de cette autorisation système.

Colonnes de la table

Nom de la colonne	Type de données	Description
nom d'utilisateur	text	Nom de l'utilisateur qui a exécuté la requête.
query	entier	ID de la requête.
xid	long	Contexte de la transaction.
pid	entier	Processus principal exécutant la requête.
recordtime	time	Heure à laquelle la requête a été enregistrée.
command	char(1)	Commande pour laquelle la politique RLS a été appliquée. Les valeurs possibles sont : `k` pour inconnu, `s` pour sélectionner, `u` pour mettre à jour, `i` pour insérer, `y` pour utilitaire, et `d` pour supprimer.

Nom de la colonne	Type de données	Description
datname	text	Nom de la base de données de la relation à laquelle la politique de sécurité au niveau des lignes est attachée.
relschema	text	Nom du schéma de la relation auquel la politique de sécurité au niveau des lignes est attachée.
relname	text	Nom de la relation auquel la politique de sécurité au niveau des lignes est attachée.
polname	text	Nom de la politique de sécurité au niveau des lignes qui est attachée à la relation.
poldefault	char(1)	Paramètre par défaut de la politique de sécurité au niveau des lignes qui est attachée à la relation. Les valeurs possibles sont : `f` pour faux si la politique fausse par défaut a été appliquée et `t` pour vrai si la politique vraie par défaut a été appliquée.

Exemple de requête

L'exemple suivant montre le résultat de SVV_RLS_APPLIED_POLICY. Pour interroger SVV_RLS_APPLIED_POLICY, vous devez disposer de l'autorisation ACCESS SYSTEM TABLE.

```
-- Check what RLS policies were applied to the run query.
SELECT username, command, datname, relschema, relname, polname, poldefault
FROM svv_rls_applied_policy
WHERE datname = CURRENT_DATABASE() AND query = PG_LAST_QUERY_ID();

 username | command |  datname  | relschema |          relname         |      polname
  | poldefault
----------+---------+-----------+-----------+--------------------------
+-----------------+------------
   molly  |    s    | tickit_db |   public  | tickit_category_redshift |
 policy_concerts |
```

SVV_RLS_ATTACHED_POLICY

Utilisez SVV_RLS_ATTACHED_POLICY pour afficher la liste de toutes les relations et de tous les utilisateurs qui disposent d'une ou de plusieurs politiques de sécurité au niveau des lignes attachées à la base de données actuellement connectée.

Seuls les utilisateurs disposant du rôle sys:secadmin peuvent interroger cette vue.

Colonnes de la table

Nom de la colonne	Type de données	Description
relschema	text	Nom du schéma de la relation auquel la politique de sécurité au niveau des lignes est attachée.
relname	text	Nom de la relation auquel la politique de sécurité au niveau des lignes est attachée.
relkind	text	Type de l'objet, tel que la table.
polname	text	Nom de la politique de sécurité au niveau des lignes qui est attachée à la relation.
grantor	text	Nom de l'utilisateur qui a attaché cette politique.
grantee	text	Nom de l'utilisateur ou du rôle auquel cette politique a été attachée.
granteekind	text	Type de bénéficiaire. Les valeurs possibles sont utilisateur ou rôle.
is_pol_on	boolean	Paramètre qui indique si une politique de sécurité au niveau des lignes est activée ou désactivée sur une table. Les valeurs possibles sont true ou false.
is_rls_on	boolean	Paramètre qui indique si une sécurité au niveau des lignes est activée ou désactivée sur une table. Les valeurs possibles sont true ou false.
rls_conjunction_type	character (3)	Paramètre qui indique si la relation combine les politiques RLS avec `and` ou `or`.

Exemple de requête

L'exemple suivant montre le résultat de SVV_RLS_ATTACHED_POLICY.

```
--Inspect the policy in SVV_RLS_ATTACHED_POLICY
SELECT * FROM svv_rls_attached_policy;

 relschema |        relname           | relkind |     polname     | grantor | grantee
 | granteekind | is_pol_on | is_rls_on | rls_conjuntion_type
-----------+--------------------------+---------+-----------------+---------+----------
+-------------+-----------+-----------+---------------------
 public    | tickit_category_redshift |  table  | policy_concerts |   bob   |  analyst
 |    role     |    True   |    True   |      and
 public    | tickit_category_redshift |  table  | policy_concerts |   bob   |  dbadmin
 |    role     |    True   |    True   |      and
```

SVV_RLS_POLICY

Utilisez SVV_RLS_POLICY pour afficher la liste de toutes les politiques de sécurité au niveau des lignes créées dans le cluster Amazon Redshift.

SVV_RLS_POLICY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
poldb	text	Nom de la base de données dans laquelle la politique de sécurité au niveau des lignes est créée.
polname	text	Nom de la politique de sécurité au niveau des lignes.
polalias	text	Alias de table utilisé dans la définition de la politique.
polatts	text	Attributs fournis à la définition de politique.
polqual	text	Condition de politique fournie dans la clause USING de l'instruction CREATE POLICY.

Nom de la colonne	Type de données	Description
polenabled	boolean	Indique si la politique est activée partout.
polmodifiedby	text	Nom de l'utilisateur qui a créé ou modifié la politique en dernier.
polmodifiedtime	timestamp	Date et heure de la création de la politique ou de la dernière modification.

Exemple de requête

L'exemple suivant montre le résultat de SVV_RLS_POLICY.

```
-- Create some policies.
CREATE RLS POLICY pol1 WITH (a int) AS t USING ( t.a IS NOT NULL );
CREATE RLS POLICY pol2 WITH (c varchar(10)) AS t USING ( c LIKE '%public%');

-- Inspect the policy in SVV_RLS_POLICY
SELECT * FROM svv_rls_policy;

 poldb | polname | polalias |                     polatts                      |
         polqual                | polenabled | polmodifiedby |   polmodifiedtime
-------+---------+----------+--------------------------------------------------
+--------------------------------------+------------+--------------
+---------------------
 my_db | pol1    | t        | [{"colname":"a","type":"integer"}]               |
 "t"."a" IS NOT NULL                   | t          | policy_admin  | 2022-02-11
 14:40:49
 my_db | pol2    | t        | [{"colname":"c","type":"character varying(10)"}] |
 "t"."c" LIKE CAST('%public%' AS TEXT) | t          | policy_admin  | 2022-02-11
 14:41:28
```

SVV_RLS_RELATION

Utilisez SVV_RLS_RELATION pour afficher la liste de toutes les relations protégées par la RLS.

SVV_RLS_RELATION est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
datname	text	Nom de la base de données contenant la relation.
relschema	text	Nom du schéma contenant la relation.
relname	text	Nom de la relation.
relkind	text	Type de relation, tel que les tables ou les vues.
is_rls_on	boolean	Paramètre qui indique si la relation est protégée par la RLS.
is_rls_datashare_on	boolean	Paramètre qui indique si la relation est protégée par RLS sur les unités de partage des données.
rls_conjunction_type	character(3)	Paramètre qui indique si la relation combine les politiques RLS avec `and` ou `or`.
rls_datashare_conjunction_type	character(3)	Paramètre qui indique si la relation combine les politiques RLS avec `and` ou `or` sur les unités de partage des données.

Exemple de requête

L'exemple suivant montre le résultat de SVV_RLS_RELATION.

```
ALTER TABLE tickit_category_redshift ROW LEVEL SECURITY ON FOR DATASHARES;


--Inspect RLS state on the relations using SVV_RLS_RELATION.
SELECT datname, relschema, relname, relkind, is_rls_on, is_rls_datashare_on FROM
 svv_rls_relation ORDER BY relname;

  datname  | relschema |        relname           | relkind | is_rls_on |
 is_rls_datashare_on | rls_conjunction_type | rls_datashare_conjunction_type
-----------+-----------+--------------------------+---------+-----------
+---------------------+----------------------+--------------------------------
```

```
 tickit_db |   public  | tickit_category_redshift |  table  |      t    |          t
       |          and         |              and
(1 row)
```

SVV_ROLE_GRANTS

Utilisez SVV_ROLE_GRANTS pour afficher la liste des rôles auxquels des rôles sont explicitement accordés dans le cluster.

SVV_ROLE_GRANTS est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les identités auxquelles ils ont accès ou qu'ils possèdent.

Colonnes de la table

Nom de la colonne	Type de données	Description
role_id	entier	ID du rôle.
role_name	text	Nom du rôle.
granted_role_id	entier	ID du rôle accordé.
granted_role_name	text	Nom du rôle accordé.

Exemple de requête

L'exemple suivant renvoie la sortie de SVV_ROLE_GRANTS..

```
GRANT ROLE role1 TO ROLE role2;
GRANT ROLE role2 TO ROLE role3;

SELECT role_name, granted_role_name FROM svv_role_grants;

 role_name |  granted_role_name
```

```
-----------+--------------------
   role2   |      role1
   role3   |      role2
(2 rows)
```

SVV_ROLES

Utilisez SVV_ROLES pour afficher les informations relatives aux rôles.

Cette table est visible pour tous les utilisateurs.

Colonnes de la table

Nom de la colonne	Type de données	Description
role_id	entier	ID du rôle.
role_name	text	Nom du rôle.
role_owner	text	Nom du propriétaire du rôle.
external_id	text	Identifiant unique du rôle dans le fournisseur d'identité tiers.

Exemple de requête

L'exemple suivant renvoie la sortie de SVV_ROLES.

```
SELECT role_name,role_owner FROM svv_roles WHERE role_name IN ('role1', 'role2');

 role_name | role_owner
-----------+------------
   role1   | superuser
   role2   | superuser
```

SVV_SCHEMA_PRIVILEGES

Utilisez SVV_SCHEMA_PRIVILEGES pour afficher les autorisations de schéma explicitement accordées aux utilisateurs, aux rôles et aux groupes de la base de données actuelle.

SVV_SCHEMA_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les identités auxquelles ils ont accès ou qu'ils possèdent.

Colonnes de la table

Nom de la colonne	Type de données	Description
namespace_name	text	Nom de l'espace de noms où existe un schéma spécifié.
privilege_type	text	Type de l'autorisation. Pour les autorisations avec un schéma privilege_scope, les valeurs possibles sont CREATE, USAGE et ALTER. Pour les valeurs privilege _scope autres que le schéma, les valeurs possibles incluent tout type d'autorisation disponible dans le champ d'application de l'autoris ation.
identity_id	entier	ID de l'identité. Les valeurs possibles sont ID d'utilisateur, ID de rôle ou ID de groupe.
identity_name	text	Nom de l'identité.
identity_type	text	Type d'identité. Les valeurs possibles sont utilisateur, rôle, groupe ou public.
admin_option	boolean	Valeur qui indique si l'utilisateur peut accorder l'autorisation à

Nom de la colonne	Type de données	Description
		d'autres utilisateurs et rôles. Elle renvoie toujours une réponse fausse pour les types d'identités rôle et groupe.
privilege_scope	text	Étendue de l'autorisation spécifiée dans privilege_type. Les valeurs possibles sont les suivantes : • SCHEMA (SCHÉMA) • TABLES • FONCTIONS Pour plus d'informations sur les autorisations délimitées, rendez-vous sur. Autorisations étendues

Exemple de requête

L'exemple suivant montre le résultat de SVV_SCHEMA_PRIVILEGES.

```
SELECT namespace_name,privilege_type,identity_name,identity_type,admin_option FROM
 svv_schema_privileges
WHERE namespace_name = 'test_schema1';

 namespace_name | privilege_type |  identity_name | identity_type | admin_option
----------------+----------------+----------------+---------------+--------------
 test_schema1   |    USAGE       |     reguser    |     user      |   False
 test_schema1   |    USAGE       |     role1      |     role      |   False
```

SVV_SCHEMA_QUOTA_STATE

Affiche le quota et l'utilisation actuelle du disque pour chaque schéma.

Les utilisateurs réguliers peuvent voir des informations sur les schémas pour lesquels ils ont une autorisation d'utilisation. Les superutilisateurs peuvent voir des informations pour tous les schémas de la base de données active.

SVV_SCHEMA_QUOTA_STATE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> Cette vue n'est disponible que lors de l'interrogation des clusters alloués.

Colonnes de la table

Nom de la colonne	Type de données	Description
schema_id	entier	ID d'espace de noms ou de schéma.
nom_schéma	caractère (128)	Nom de l'espace de noms ou du schéma.
schema_owner	entier	ID utilisateur interne du propriétaire du schéma.
quota	entier	Quantité d'espace disque (en Mo) que le schéma peut utiliser.
disk_usage	entier	Espace disque (en Mo) actuellem ent utilisé par le schéma.
disk_usage_pct	double precision	Pourcentage d'espace disque actuellement utilisé par le schéma hors du quota configuré.

Exemple de requête

L'exemple suivant montre comment afficher le quota et l'utilisation actuelle du disque pour le schéma.

```
SELECT TRIM(SCHEMA_NAME) "schema_name", QUOTA, disk_usage, disk_usage_pct FROM
 svv_schema_quota_state
WHERE SCHEMA_NAME = 'sales_schema';
schema_name   | quota | disk_usage | disk_usage_pct
--------------+-------+------------+----------------
sales_schema  | 2048  | 30         | 1.46
(1 row)
```

SVV_SYSTEM_PRIVILEGES

SVV_SYSTEM_PRIVILEGES est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les identités auxquelles ils ont accès ou qu'ils possèdent.

Colonnes de la table

Nom de la colonne	Type de données	Description
system_privilege	text	Nom de l'autorisation système.
identity_id	entier	ID de l'identité. Les valeurs possibles sont ID d'utilisateur ou ID de rôle.
identity_name	text	Nom de l'identité.
identity_type	text	Type d'identité. Les valeurs possibles sont utilisateur ou rôle.

Exemple de requête

L'exemple suivant affiche le résultat pour les paramètres spécifiés.

```
SELECT system_privilege,identity_name,identity_type FROM svv_system_privileges
WHERE system_privilege = 'ALTER TABLE' AND identity_name = 'sys:superuser';

 system_privilege | identity_name | identity_type
------------------+---------------+---------------
   ALTER TABLE    | sys:superuser |     role
```

SVV_TABLE_INFO

Affiche les informations récapitulatives des tables de la base de données. La vue filtre les tables système et affiche uniquement les tables définies par l'utilisateur.

Vous pouvez utiliser la vue SVV_TABLE_INFO pour diagnostiquer et traiter les problèmes de conception de table qui peuvent influencer les performances des requêtes. Cela inclut les problèmes d'encodage de compression, les clés de distribution, le style de tri, l'asymétrie de la distribution des données, la taille de la table et les statistiques. La vue SVV_TABLE_INFO ne renvoie pas d'informations pour les tables vides.

La vue SVV_TABLE_INFO récapitule les informations des tables système STV_BLOCKLIST, STV_NODE_STORAGE_CAPACITY, STV_TBL_PERM et STV_SLICES, ainsi que des tables de catalogue PG_DATABASE, PG_ATTRIBUTE, PG_CLASS, PG_NAMESPACE et PG_TYPE.

SVV_TABLE_INFO n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système. Pour permettre à un utilisateur d'interroger la vue, accordez-lui l'autorisation SELECT sur SVV_TABLE_INFO.

Colonnes de la table

Nom de la colonne	Type de données	Description
database	text	Nom de la base de données.
schema	text	Nom du schéma.
table_id	oid	ID de table.

Nom de la colonne	Type de données	Description
table	text	Nom de la table.
encoded	text	Valeur qui indique si une colonne possède un encodage de compression défini.
diststyle	text	Style de distribution ou colonne de clé de distribut ion, si la distribution de clés est définie. Les valeurs possibles incluent EVEN, KEY(*column*), ALL, AUTO(ALL) , AUTO(EVEN) et AUTO(KEY(*column*)).
sortkey1	text	Première colonne de la clé de tri, si une clé de tri est définie. Les valeurs possibles incluent *column*, AUTO(SORTKEY) et AUTO(SORTKEY(*column*)).
max_varchar	entier	Taille de la plus grande colonne qui utilise un type de données VARCHAR.
sortkey1_enc	character(32)	Encodage de compression de la première colonne de la clé de tri, si une clé de tri est définie.
sortkey_num	entier	Nombre de colonnes définies comme clés de tri.
size	bigint	Taille de la table, en blocs de données de 1 Mo.

Nom de la colonne	Type de données	Description
pct_used	numeric(10,4)	Pourcentage de l'espace disponible utilisé par la table.
empty	bigint	Pour utilisation interne. Cette colonne est n'est plus utilisée et sera supprimée dans une version ultérieure.
unsorted	numeric(5,2)	Pourcentage de lignes non triées de la table.
stats_off	numeric(5,2)	Nombre qui indique le degré d'obsolescence des statistiq ues de la table ; 0 indique des statistiques à jour, 100 des statistiques obsolètes.
tbl_rows	numeric(38,0)	Nombre total de lignes de la table. Cette valeur inclut les lignes marquées pour la suppression, mais pas encore aspirées.
skew_sortkey1	numeric(19,2)	Rapport entre la taille de la colonne de clé autre que la clé de tri la plus importante et la taille de la première colonne de la clé de tri, si une clé de tri est définie. Utilisez cette valeur pour évaluer l'efficacité de la clé de tri.

Nom de la colonne	Type de données	Description
skew_rows	numeric(19,2)	Rapport entre le nombre de lignes de la tranche avec le plus de lignes et le nombre de lignes de la tranche avec le moins de lignes.
estimated_visible_rows	numeric(38,0)	Estimation du nombre de ligne de la table. Cette valeur n’inclut pas les lignes marquées pour la suppressi on.

Nom de la colonne	Type de données	Description

risk_event	text	Informations sur les risques d'une table. Le champ est séparé en deux parties :
		
		`risk_type |xid|timestamp`
		
		• Le `risk_type` , où `1` indique l'exécution d'une `COPY command with the EXPLICIT_IDS option` . Amazon Redshift ne vérifie plus l'unicité des colonnes IDENTITY dans la table. Pour plus d'informations, consultez EXPLICIT_IDS.
		• L'ID de transaction, `xid`, qui introduit le risque.
		• Le `timestamp` lors de l'exécution de la commande COPY.
		
		L'exemple suivant présente les valeurs du champ.
		
		`1|1107|2019-06-22 07:16:11.292952`

vacuum_sort_benefit	numeric(12,2)	Pourcentage maximum estimé d'amélioration de la performance des requêtes d'analyse lorsque vous exécutez une opération VACUUM SORT.

Nom de la colonne	Type de données	Description
create_time	horodatage sans fuseau horaire	L'heure à laquelle la table a été créée.

Exemples de requêtes

L'exemple suivant affiche l'encodage, le style de distribution, le tri et le delta des données de toutes les tables de la base de données définies par l'utilisateur. Ici, "table" doit être entre guillemets doubles, car il s'agit d'un mot réservé.

```
select "table", encoded, diststyle, sortkey1, skew_sortkey1, skew_rows
from svv_table_info
order by 1;

table          | encoded | diststyle       | sortkey1     | skew_sortkey1 | skew_rows
---------------+---------+-----------------+--------------+---------------+----------
category       | N       | EVEN            |              |               |
date           | N       | ALL             | dateid       |          1.00 |
event          | Y       | KEY(eventid)    | dateid       |          1.00 |      1.02
listing        | Y       | KEY(listid)     | dateid       |          1.00 |      1.01
sales          | Y       | KEY(listid)     | dateid       |          1.00 |      1.02
users          | Y       | KEY(userid)     | userid       |          1.00 |      1.01
venue          | N       | ALL             | venueid      |          1.00 |
(7 rows)
```

SVV_TABLES

Utilisez SVV_TABLES pour afficher les tables dans des catalogues locaux et externes.

SVV_TABLES est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
table_catalog	text	Nom du catalogue contenant la table.
table_schema	text	Nom du schéma de la table.
table_name	text	Nom de la table.
table_type	text	Type de la table. Les valeurs possibles sont les vues, les tables externes et les tables de base.
remarks	text	Remarques.

SVV_TRANSACTIONS

Enregistre les informations sur les transactions qui maintiennent actuellement des verrous sur les tables de la base de données. Utilisez la vue SVV_TRANSACTIONS pour identifier les transactions en cours et les problèmes de conflit de verrous. Pour plus d'informations sur les verrous, consultez Gestion des opérations d'écriture simultanées et LOCK.

SVV_TRANSACTIONS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
txn_owner	text	Nom du propriétaire de la transaction.
txn_db	text	Nom de la base de données associée à la transaction.

Nom de la colonne	Type de données	Description
xid	bigint	ID de transaction.
pid	entier	ID de processus associé au verrou.
txn_start	timestamp	Heure de début de la transacti on.
lock_mode	text	Nom du mode de verrou maintenu ou demandé par ce processus. Si `lock_mode` est `ExclusiveLock` et que `granted` a la valeur true (`t`), cet ID de transaction est une transaction ouverte.
lockable_object_type	text	Type d'objet demandant ou maintenant le verrou, soit `relation` si c'est une table, soit `transactionid` si c'est une transaction.
relation	entier	ID de table de la table (relation) obtenant le verrou. Cette valeur est NULL si `lockable_object_type` est `transactionid` .
granted	boolean	Valeur qui indique si le verrou a été accordé (`t`) ou s'il est en attente (`f`).

Exemples de requêtes

La commande suivante affiche toutes les transactions actives et les verrous demandés par chaque transaction.

```
select * from svv_transactions;

 txn_
 lockable_
 owner | txn_db |  xid   |  pid  |         txn_start          |      lock_mode      |
 object_type    | relation | granted
-------+--------+--------+-------+----------------------------+---------------------
+---------------+----------+---------
 root  | dev    | 438484 | 22223 | 2016-03-02 18:42:18.862254 | AccessShareLock     |
 relation       |   100068 | t
 root  | dev    | 438484 | 22223 | 2016-03-02 18:42:18.862254 | ExclusiveLock       |
 transactionid  |          | t
 root  | tickit | 438490 | 22277 | 2016-03-02 18:42:48.084037 | AccessShareLock     |
 relation       |    50860 | t
 root  | tickit | 438490 | 22277 | 2016-03-02 18:42:48.084037 | AccessShareLock     |
 relation       |    52310 | t
 root  | tickit | 438490 | 22277 | 2016-03-02 18:42:48.084037 | ExclusiveLock       |
 transactionid  |          | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100068 | f
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | RowExclusiveLock    |
 relation       |    16688 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessShareLock     |
 relation       |   100064 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100166 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100171 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | AccessExclusiveLock |
 relation       |   100190 | t
 root  | dev    | 438505 | 22378 | 2016-03-02 18:43:27.611292 | ExclusiveLock       |
 transactionid  |          | t
(12 rows)

(12 rows)
```

SVV_USER_GRANTS

Utilisez SVV_USER_GRANTS pour afficher la liste des utilisateurs auxquels des rôles sont explicitement accordés dans le cluster.

SVV_USER_GRANTS est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant de l'autorisation ACCESS SYSTEM TABLE

Les autres utilisateurs ne peuvent consulter que les rôles qui leur sont explicitement accordés.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	ID de l'utilisateur.
user_name	text	Le nom de l'utilisateur.
role_id	entier	ID de rôle du rôle accordé.
role_name	text	Nom de rôle du rôle accordé.
admin_option	boolean	Valeur qui indique si l'utilisateur peut accorder le rôle à d'autres utilisateurs et rôles.

Exemples de requêtes

Les requêtes suivantes accordent des rôles aux utilisateurs et affichent la liste des utilisateurs auxquels des rôles sont explicitement accordés.

```
GRANT ROLE role1 TO reguser;
GRANT ROLE role2 TO reguser;
GRANT ROLE role1 TO superuser;
GRANT ROLE role2 TO superuser;

SELECT user_name,role_name,admin_option FROM svv_user_grants;

 user_name | role_name | admin_option
-----------+-----------+--------------
 superuser |  role1    | False
 reguser   |  role1    | False
 superuser |  role2    | False
  reguser  |  role2    | False
```

SVV_USER_INFO

Vous pouvez récupérer des données sur les utilisateurs de la base de données Amazon Redshift avec la vue SVV_USER_INFO.

SVV_USER_INFO est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_name	text	Nom d'utilisateur pour le rôle.
user_id	entier	ID de l'utilisateur.
createdb	boolean	Valeur qui indique si l'utilisateur dispose des autorisations pour créer des bases de données.
super-utilisateur	boolean	Valeur qui indique si l'utilisateur est un super-utilisateur.
catalog_update	boolean	Valeur qui indique si l'utilisateur peut mettre à jour les catalogues système.
connection_limit	text	Nombre de connexions que l'utilisateur peut ouvrir.
syslog_access	text	Valeur qui indique si l'utilisateur a accès aux journaux système. Les deux valeurs possibles sont `RESTRICTED` et `UNRESTRICTED`. `RESTRICTED` signifie que les utilisateurs qui ne sont pas des super-utilisateurs peuvent voir leurs propres enregistrements. `UNRESTRICTED` signifie que les utilisateurs qui ne sont pas des super-utilisateurs peuvent voir tous les enregistrements dans les tables et les vues système pour lesquelles ils disposent de privilèges `SELECT`.

Nom de la colonne	Type de données	Description
last_ddl_timestamp	timestamp	Horodatage de la dernière instruction de création de langage de définition de données (DDL) exécutée par l'utilisateur.
session_timeout	entier	La durée maximale d'inactivité d'une séance avant son expiration (exprimée en secondes). 0 indique qu'aucun délai d'expiration n'a été défini. Pour plus d'informations sur le paramètre de délai d'inactivité du cluster, consultez Quotas et limites d'Amazon Redshift du guide de gestion des clusters Amazon Redshift.
external_user_id	text	Identifiant unique de l'utilisateur dans le fournisseur d'identité tiers.

Exemples de requêtes

La commande suivante récupère les informations d'utilisateur à partir de SVV_USER_INFO.

```
SELECT * FROM SVV_USER_INFO;
```

SVV_VACUUM_PROGRESS

Cette vue renvoie une estimation de la durée nécessaire à l'exécution d'une opération VACUUM en cours.

SVV_VACUUM_PROGRESS n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_VACUUM_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Pour plus d'informations sur SVV_VACUUM_SOMM_SUMMY, consultez SVV_VACUUM_SUMMARY.

Pour plus d'informations sur SVL_VACUUM_PERCENT, consultez SVL_VACUUM_PERCENTAGE.

> **Note**
>
> Cette vue n'est disponible que lors de l'interrogation des clusters alloués.

Colonnes de la table

Nom de la colonne	Type de données	Description
table_name	text	Nom de la table actuellement en cours d'aspiration (VACUUM) ou de la dernière table aspirée (VACUUM) si aucune opération n'est en cours.
État	text	Description de l'activité en cours exécutée dans le cadre de l'opération VACUUM : • Initialiser • Tri • Fusionner • Suppression • Select • Échec • Complet • Ignoré • Génération de l'ordre INTERLEAVED SORTKEY
time_remaining_estimate	text	Temps restant estimé pour que l'opération VACUUM en cours se termine, exprimé en minutes et en secondes : **`5m 10s`**, par exemple. Une estimation n'est pas retournée tant que l'opération VACUUM n'a pas terminé sa première opération de tri. Si aucune opération VACUUM n'est en cours, la dernière opération VACUUM exécutée s'affiche avec **`Completed`** dans la colonne STATUS et une colonne TIME_REMAINING_ESTIMATE vide. L'estimation devient généralement

Nom de la colonne	Type de données	Description
		plus précise au fur et à mesure que l'opération VACUUM progresse.

Exemples de requêtes

Les requêtes suivantes, exécutées à quelques minutes d'intervalle, montrent qu'une grande table nommée SALESNEW est en cours d'aspiration.

```
select * from svv_vacuum_progress;

table_name    |            status             | time_remaining_estimate
--------------+-------------------------------+------------------------
salesnew      |  Vacuum: initialize salesnew  |
(1 row)
...
select * from svv_vacuum_progress;

table_name   |         status         | time_remaining_estimate
-------------+------------------------+------------------------
salesnew     |  Vacuum salesnew sort  | 33m 21s
(1 row)
```

La requête suivante affiche qu'aucune opération d'aspiration n'est actuellement en cours. La dernière table à avoir été aspirée était la table SALES.

```
select * from svv_vacuum_progress;

table_name   |  status  | time_remaining_estimate
-------------+----------+------------------------
  sales      | Complete |
(1 row)
```

SVV_VACUUM_SUMMARY

La vue SVV_VACUUM_SUMMARY joint les tables STL_VACUUM, STL_QUERY et STV_TBL_PERM pour résumer les informations sur les opérations d'aspiration enregistrées par le système. La vue renvoie une ligne par table et par transaction VACUUM. La vue enregistre la durée de l'opération,

le nombre de partitions de tri créées, le nombre d'incréments de fusion requis et les différences en nombres de ligne et de bloc avant et après l'exécution de l'opération.

SVV_VACUUM_SUMMARY n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_VACUUM_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Pour plus d'informations sur SVV_VACUUM_PROGRESS, consultez SVV_VACUUM_PROGRESS.

Pour plus d'informations sur SVL_VACUUM_PERCENT, consultez SVL_VACUUM_PERCENTAGE.

> **Note**
>
> Cette vue n'est disponible que lors de l'interrogation des clusters alloués.

Colonnes de la table

Nom de la colonne	Type de données	Description
table_name	text	Nom de la table aspirée.
xid	bigint	ID de transaction de l'opération VACUUM.
sort_partitions	bigint	Nombre de partitions triées créées lors de la phase de tri de l'opération VACUUM.
merge_increments	bigint	Nombre d'incréments de fusion requis pour terminer la phase de fusion de l'opération VACUUM.
elapsed_time	bigint	Délai d'exécution de l'opération VACUUM (en microseco ndes).
row_delta	bigint	Différence dans le nombre total de lignes de table avant et après l'opération VACUUM.

Nom de la colonne	Type de données	Description
sortedrow_delta	bigint	Différence dans le nombre de lignes de table triées avant et après l'opération VACUUM.
block_delta	entier	Différence dans le nombre de blocs de la table avant et après l'opération VACUUM.
max_merge_partitions	entier	Cette colonne est utilisée pour l'analyse des performances et représente le nombre maximal de partitions qu'une opération VACUUM peut traiter pour la table par itération de phase de fusion. (L'opération VACUUM trie la région non triée en une ou plusieurs partitions. Selon le nombre de colonnes de la table et la configuration actuelle d'Amazon Redshift, la phase de fusion peut traiter un nombre maximal de partitions en une seule itération de fusion. La phase de fusion continue si le nombre de partitions triées dépasse le nombre maximal de partitions de fusion, mais un plus grand nombre d'itérations de fusion sera nécessaire.)

Exemple de requête

La requête suivante renvoie les statistiques des opérations VACUUM sur trois tables différentes. La table SALES a été aspirée deux fois.

```
select table_name, xid, sort_partitions as parts, merge_increments as merges,
elapsed_time, row_delta, sortedrow_delta as sorted_delta, block_delta
from svv_vacuum_summary
order by xid;

table_  | xid  |parts|merges| elapsed_ | row_    | sorted_ | block_
name    |      |     |      | time     | delta   | delta   | delta
--------+------+-----+------+----------+---------+---------+--------
users   | 2985 |   1 |    1 | 61919653 |       0 |   49990 |      20
category| 3982 |   1 |    1 | 24136484 |       0 |      11 |       0
sales   | 3992 |   2 |    1 | 71736163 |       0 | 1207192 |      32
sales   | 4000 |   1 |    1 | 15363010 | -851648 | -851648 |    -140
```

```
(4 rows)
```

Vues de surveillance SYS

Les vues de surveillance sont des vues système dans Amazon Redshift qui permettent de surveiller l'utilisation de ressources de requête et de charge de travail par les clusters mis en service et les groupes de travail sans serveur. Ces vues se trouvent dans le schéma `pg_catalog`. Pour afficher les informations fournies par ces vues, exécutez les instructions SQL SELECT.

Sauf indication contraire, ces vues sont disponibles pour les clusters Amazon Redshift et les groupes de travail Amazon Redshift sans serveur.

SYS_SERVERLESS_USAGE collecte les données d'utilisation pour Amazon Redshift sans serveur uniquement.

Rubriques

- SYS_ANALYZE_COMPRESSION_HISTORY
- SYS_ANALYZE_HISTORY
- SYS_APPLIED_MASKING_POLICY_LOG
- OPTIMISATION DE LA TABLE SYS_AUTOMATIQUE
- SYS_CHILD_QUERY_TEXT
- SYS_CONNECTION_LOG
- SYS_COPY_JOB
- SYS_COPY_JOB_DETAIL
- SYS_COPY_JOB_INFO
- SYS_COPY_REPLACEMENTS
- SYS_DATASHARE_CHANGE_LOG
- SYS_DATASHARE_CROSS_REGION_USAGE
- SYS_DATASHARE_USAGE_CONSUMER
- SYS_DATASHARE_USAGE_PRODUCER
- SYS_EXTERNAL_QUERY_DETAIL
- SYS_EXTERNAL_QUERY_ERROR
- SYS_INTEGRATION_ACTIVITY
- SYS_INTEGRATION_TABLE_ACTIVITY

- SYS_INTEGRATION_TABLE_STATE_CHANGE
- SYS_LOAD_DETAIL
- SYS_LOAD_ERROR_DETAIL
- SYS_LOAD_HISTORY
- SYS_MV_REFRESH_HISTORY
- SYS_MV_STATE
- SYS_PROCEDURE_CALL
- SYS_PROCEDURE_MESSAGES
- SYS_QUERY_DETAIL
- SYS_QUERY_EXPLAIN
- SYS_QUERY_HISTORY
- SYS_QUERY_TEXT
- SYS_RESTORE_LOG
- SYS_RESTORE_STATE
- SYS_SCHEMA_QUOTA_VIOLATIONS
- SYS_SERVERLESS_USAGE
- SYS_SESSION_HISTORY
- SYS_SPATIAL_SIMPLIFY
- SYS_STREAM_SCAN_ERRORS
- SYS_STREAM_SCAN_STATES
- SYS_TRANSACTION_HISTORY
- SYS_UDF_LOG
- SYS_UNLOAD_DETAIL
- SYS_UNLOAD_HISTORY
- SYS_USERLOG
- SYS_VACUUM_HISTORY

SYS_ANALYZE_COMPRESSION_HISTORY

Enregistre les détails pour les opérations d’analyse de compression au cours de commandes COPY ou ANALYZE COMPRESSION.

SYS_ANALYZE_COMPRESSION_HISTORY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	ID de l'utilisateur qui a généré l'entrée.
start_time	timestamp	Heure à laquelle l'opération d'analyse de compression a commencé.
transaction_id	bigint	ID de transaction de l'opération d'analyse de compression.
table_id	entier	ID de la table analysée.
table_name	character(128)	Nom de la table analysée.
column_position	entier	Index de la colonne de la table analysée pour déterminer l'encodage de compression.
old_encoding	character(15)	Type d'encodage avant l'analyse de compression.
new_encoding	character(15)	Type d'encodage après l'analyse de compression.
mode	character(14)	Les valeurs possibles sont : PRESET Spécifie que `new_encoding` est déterminé par la commande COPY Amazon Redshift en fonction du type de données de la colonne. Aucune donnée échantillonnée.

Nom de la colonne	Type de données	Description
		ON Spécifie que `new_encoding` est déterminé par la commande COPY Amazon Redshift en fonction de l'analyse d'un exemple de données. ANALYZE ONLY Spécifie que `new_encoding` est déterminé par la commande ANALYZE COMPRESSION Amazon Redshift en fonction de l'analyse d'un exemple de données. Toutefois, le type d'encodage de la colonne analysée n'est pas modifié.

Exemples de requêtes

L'exemple suivant examine les détails de l'analyse de compression effectuée sur la table `lineitem` par la dernière commande COPY exécutée dans la même séance.

```
select transaction_id, table_id, btrim(table_name) as table_name, column_position,
 old_encoding, new_encoding, mode
from sys_analyze_compression_history
where transaction_id = (select transaction_id from sys_query_history where query_id =
 pg_last_copy_id()) order by column_position;

 transaction_id  |  table_id   | table_name | column_position |  old_encoding   |
 new_encoding   |      mode
-----------------+-------------+------------+-----------------+-----------------
+-----------------+-------------
      8196       |   248126    | lineitem   |        0        | mostly32        |
 mostly32        | ON
      8196       |   248126    | lineitem   |        1        | mostly32        | lzo
           | ON
      8196       |   248126    | lineitem   |        2        | lzo             |
 delta32k        | ON
      8196       |   248126    | lineitem   |        3        | delta           | delta
           | ON
```

```
      8196       |   248126    | lineitem   |        4        | bytedict        |
 bytedict        | ON
      8196       |   248126    | lineitem   |        5        | mostly32        |
 mostly32        | ON
      8196       |   248126    | lineitem   |        6        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |        7        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |        8        | lzo             | zstd
           | ON
      8196       |   248126    | lineitem   |        9        | runlength       | zstd
           | ON
      8196       |   248126    | lineitem   |       10        | delta           | lzo
           | ON
      8196       |   248126    | lineitem   |       11        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |       12        | delta           | delta
           | ON
      8196       |   248126    | lineitem   |       13        | bytedict        | zstd
           | ON
      8196       |   248126    | lineitem   |       14        | bytedict        | zstd
           | ON
      8196       |   248126    | lineitem   |       15        | text255         | zstd
           | ON
(16 rows)
```

SYS_ANALYZE_HISTORY

Journalise les détails des opérations ANALYZE.

SYS_ANALYZE_HISTORY n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	ID de l'utilisateur qui a généré l'entrée.
transaction_id	long	ID de transaction.

Nom de la colonne	Type de données	Description
database_name	char(30)	Nom de la base de données.
table_name	char(30)	Nom de la table.
table_id	entier	ID de la table.
is_automatic	char(1)	La valeur est true (t) si l'opération incluait une opération ANALYZE Amazon Redshift par défaut. La valeur est false (f) si la commande ANALYZE a été exécutée explicitement.
status	char(15)	Résultat de la commande d'analyse. Les valeurs possibles sont Full, Skipped et PredicateColumn.
start_time	timestamp	Heure UTC à laquelle l'opération ANALYZE a commencé à s'exécuter.
end_time	timestamp	Heure UTC à laquelle l'opération ANALYZE a fini de s'exécuter.
rows	double	Nombre total de lignes de la table
modified_rows	double	Nombre total de lignes qui ont été modifiées depuis la dernière opération ANALYZE.
analyze_threshold_percent	entier	Valeur du paramètre analyze_threshold_percent.
last_analyze_time	timestamp	Heure UTC à laquelle la table a été analysée précédemment.

Exemples de requêtes

```
 user_id | transaction_id | database_name | schema_name |      table_name     |
 table_id | is_automatic | Status |         start_time         |          end_time
     | rows | modified_rows | analyze_threshold_percent |  last_analyze_time
---------+----------------+---------------+------------+--------------------
+----------+--------------+--------+----------------------------+----------
+-----------------+------+--------------+--------------------------
+---------------------
     101 |           8006 |           dev |      public | test_table_562bf8dc
 |   110427 |            f |   Full | 2023-09-21 18:33:08.504646 | 2023-09-21
 18:33:24.296498 |    5 |             5 |                          0 | 2000-01-01
 00:00:00
```

SYS_APPLIED_MASKING_POLICY_LOG

Utilisez SYS_APPLIED_MASKING_POLICY_LOG pour suivre l'application des politiques de masquage dynamique des données aux requêtes qui font référence à des relations protégées par DDM.

SYS_APPLIED_MASKING_POLICY_LOG est visible par les utilisateurs suivants :

- Super-utilisateurs
- Utilisateurs disposant du rôle `sys:operator`
- Utilisateurs disposant de l’autorisation ACCESS SYSTEM TABLE

Les utilisateurs réguliers verront 0 ligne.

Notez que SYS_APPLIED_MASKING_POLICY_LOG n'est pas visible pour les utilisateurs dotés du rôle. `sys:secadmin`

Pour plus d'informations sur le masquage dynamique des données, rendez-vous sur. Masquage dynamique des données

Colonnes de la table

Nom de la colonne	Type de données	Description
policy_name	text	Nom de la politique de masquage.

Nom de la colonne	Type de données	Description
user_id	text	ID de l'utilisateur qui a exécuté la requête.
record_time	timestamp	Heure à laquelle l'entrée de la vue système a été enregistrée.
session_id	int	ID du processus.
transaction_id	long	ID de transaction.
query_id	int	ID de requête.
database_name	text	Nom de la base de données sur laquelle la requête a été exécutée.
relation_name	text	Nom de la table à laquelle la politique de masquage est appliquée.
nom_schéma	text	Nom du schéma dans lequel se trouve la table.
identifiant_pièce jointe	long	L'identifiant de la politique de masquage ci-jointe.
type de relation	text	Type de relation à laquelle la politique de masquage est appliquée. Les valeurs possibles sont `TABLE`, `VIEW`, `LATE BINDING VIEW` et `MATERIALIZED VIEW` .

Exemples de requêtes

L'exemple suivant montre que la politique de `mask_credit_card_full` masquage est attachée à la `credit_db.public.credit_cards` table.

```
select policy_name, database_name, relation_name, schema_name, relation_kind
from sys_applied_masking_policy_log;

policy_name           | database_name | relation_name | schema_name | relation_kind
```

```
----------------------+--------------+---------------+-------------+---------------
mask_credit_card_full | credit_db    | credit_cards  | public      | table

(1 row)
```

OPTIMISATION DE LA TABLE SYS_AUTOMATIQUE

Enregistre les actions automatisées d'Amazon Redshift sur les tables définies pour l'optimisation automatique.

SYS_AUTO_TABLE_OPTIMIZATION n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
transaction_id	long	Identificateur de transaction.
session_id	int	Identifiant de session du processus qui a exécuté la commande alter.
table_id	int	L'identificateur de table.
alter_type de table	character (32)	Le type de recommandation. Les valeurs possibles sont `distkey`, `sortkey` et `encode`.
status	character (128)	Le statut d'achèvement de la recommandation. Les valeurs possibles sont `Start`, `Complete`, `Skipped`, `Abort`, `Checkpoint` et `Failed`.
event_time	timestamp	Horodatage de la colonne d'état.
alter_from	character (200)	Le style de distribution précédent et les clés de tri de la table avant d'appliquer la recommandation. La valeur est tronquée en incréments de 200 caractères.

Exemples de requêtes

Dans l'exemple suivant, les lignes de résultats affichent les actions d'Amazon Redshift.

```
SELECT table_id, alter_table_type, status, event_time, alter_from
FROM SYS_AUTO_TABLE_OPTIMIZATION;

 table_id |  alter_table_type   |                        status
 |         event_time          |  alter_from
----------+---------------------+-----------------------------------------------------
+-----------------------------+-----------------
   118082 | sortkey             | Start
 | 2020-08-22 19:42:20.727049  |
   118078 | sortkey             | Start
 | 2020-08-22 19:43:54.728819  |
   118082 | sortkey             | Start
 | 2020-08-22 19:42:52.690264  |
   118072 | sortkey             | Start
 | 2020-08-22 19:44:14.793572  |
   118082 | sortkey             | Failed
 | 2020-08-22 19:42:20.728917  |
   118078 | sortkey             | Complete
 | 2020-08-22 19:43:54.792705  |  SORTKEY: None;
   118086 | sortkey             | Complete
 | 2020-08-22 19:42:00.72635   |  SORTKEY: None;
   118082 | sortkey             | Complete
 | 2020-08-22 19:43:34.728144  |  SORTKEY: None;
   118072 | sortkey             | Skipped:Retry exceeds the maximum limit for a table.
 | 2020-08-22 19:44:46.706155  |
   118086 | sortkey             | Start
 | 2020-08-22 19:42:00.685255  |
   118082 | sortkey             | Start
 | 2020-08-22 19:43:34.69531   |
   118072 | sortkey             | Start
 | 2020-08-22 19:44:46.703331  |
   118082 | sortkey             | Checkpoint: progress 14.755079%
 | 2020-08-22 19:42:52.692828  |
   118072 | sortkey             | Failed
 | 2020-08-22 19:44:14.796071  |
   116723 | sortkey             | Abort:This table is not AUTO.
 | 2020-10-28 05:12:58.479233  |
   110203 | distkey             | Abort:This table is not AUTO.
 | 2020-10-28 05:45:54.67259   |
```

SYS_CHILD_QUERY_TEXT

Renvoie le texte SQL d'une requête enfant.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identificateur de l'utilisateur qui a envoyé la requête.
query_id	bigint	L'ID de requête de l'utilisateur
child_query_séquence	entier	La séquence de la requête utilisateur réécrite, en commençant par 1.
sequence	entier	Numéro de séquence pour cet élément de requête.
text	character (200)	Les 200 premiers caractères du texte de la requête SQL.

Exemples de requêtes

Dans l'exemple suivant, les lignes de résultats affichent les actions d'Amazon Redshift.

```
SELECT * from sys_child_query_text where query_id = '34487366' order by
 child_query_sequence asc, sequence asc;

user_id | query_id | child_query_sequence | sequence | text
--------|----------|----------------------|----------|-----------------------------------------
100     | 34899339 |   1                  |    0     |  /* RQEV2-aY6ZZ1ZpQK */\nwith
 venue as (\n    select venueid,\n            venuename,\n            venuestate\n
 from venue\n), event as (\n    select eventid,\n            venueid,\n            date

100     | 34899339 |   1                  |    1     |  id,\n            eventname\n
  from event\n    where eventname like '3 Doors Down'\n), users as (\n    select userid
\n    from users\n), sales as (\n    select salesid,\n            pricepaid,
100     | 34899339 |   1                  |    2     |  \n            eventid,\n
       buyerid\n    from sales\n)\nselect e.eventname,\n        v.venuename,\n
 count(distinct(u.userid)) as unique_customers,\n        sum(s.pricepaid) as total_sal

100     | 34899339 |   1                  |    3     |  es\nfrom venue as v inner join
 event e on v.venueid = e.venueid\ninner join sales s on e.eventid = s.eventid inner
```

```
join users u on s.buyerid = u.userid\ngroup by 1,2\norder by 4 desc limit 100
```

SYS_CONNECTION_LOG

Enregistre les tentatives d'authentification, ainsi que les connexions et déconnexions.

SYS_CONNECTION_LOG n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
event	character(50)	Connexion ou événement d'authentification.
record_time	timestamp	Heure de l'événement.
remote_host	character(45)	Nom ou adresse IP de l'hôte distant.
remote_port	character(32)	Numéro de port de l'hôte distant.
session_id	entier	ID de processus associé à l'instruction.
database_name	character(50)	Nom de la base de données.
user_name	character(50)	Nom d'utilisateur.
auth_method	character(32)	Méthode d'authentification.
duration	entier	Durée de connexion en microsecondes.
ssl_version	character(50)	Version du protocole SSL (Secure Sockets Layer). Cette colonne est vide lorsque le protocole SSL est désactivé.
ssl_cipher	character(128)	Chiffrement SSL.

Nom de la colonne	Type de données	Description
mtu	entier	Unité de transmission maximale (MTU).
ssl_compression	character(64)	Type de compression SSL.
ssl_expansion	character(64)	Type d’extension SSL.
iam_auth_guid	character(36)	ID d'authentification IAM pour la CloudTrail demande.
application_name	character(250)	Nom initial ou mis à jour de l’application pour une séance.
driver_version	character(64)	La version du pilote ODBC ou JDBC qui se connecte à votre cluster Amazon Redshift à partir de vos outils clients SQL tiers.
os_version	character(64)	La version du système d’exploitation de la machine cliente qui se connecte à votre cluster Amazon Redshift.
plugin_name	character(32)	Le nom du plugin utilisé pour se connecter à votre cluster Amazon Redshift.

Nom de la colonne	Type de données	Description
protocol_version	entier	La version du protocole interne que le pilote Amazon Redshift utilise pour établir sa connexion avec le serveur. Les versions du protocole sont négociées entre le pilote et le serveur. La version décrit les fonctions disponibles. Les valeurs valides sont les suivantes : • 0 (BASE_SERVER_PROTOCOL_VERSION) • 1 (EXTENDED_RESULT_METADATA_SERVER_PROTOCOL_VERSION) – Pour enregistrer un aller-retour par requête, le serveur envoie des informations supplémentaires sur les métadonnées du jeu de résultats. • 2 (BINARY_PROTOCOL_VERSION) – Selon le type de données du jeu de résultats, le serveur envoie des données au format binaire. • 3 (EXTENDED2_RESULT_METADATA_SERVER_PROTOCOL_VERSION) — Le serveur envoie les informations de distinction majuscules/minuscules (classement) d'une colonne.
global_session_id	character(36)	Identifiant unique au niveau mondial pour la séance en cours. L'ID de séance persiste lors des redémarrages échoués du nœud.

Exemples de requêtes

Pour afficher les détails pour les connexions ouvertes, exécutez la requête suivante.

```
select record_time, user_name, database_name, remote_host, remote_port
from sys_connection_log
where event = 'initiating session'
and session_id not in
(select session_id from sys_connection_log
where event = 'disconnecting session')
order by 1 desc;
```

```
record_time         | user_name   | database_name   | remote_host   | remote_port

--------------------+-------------+-----------------+---------------
+--------------------------------
2014-11-06 20:30:06 | rdsdb       | dev             | [local]       |

2014-11-06 20:29:37 | test001     | test            | 10.49.42.138  | 11111

2014-11-05 20:30:29 | rdsdb       | dev             | 10.49.42.138  | 33333

2014-11-05 20:28:35 | rdsdb       | dev             | [local]       |
(4 rows)
```

L'exemple suivant reflète un échec de tentative d'authentification, et une connexion et une déconnexion réussie.

```
select event, record_time, remote_host, user_name
from sys_connection_log order by record_time;

            event      |         record_time        |  remote_host  | user_name

-----------------------+----------------------------+---------------+---------
authentication failure | 2012-10-25 14:41:56.96391  | 10.49.42.138  | john

authenticated          | 2012-10-25 14:42:10.87613  | 10.49.42.138  | john

initiating session     | 2012-10-25 14:42:10.87638  | 10.49.42.138  | john

disconnecting session  | 2012-10-25 14:42:19.95992  | 10.49.42.138  | john

(4 rows)
```

L'exemple suivant montre la version du pilote ODBC, le système d'exploitation de la machine cliente et le plugin utilisé pour se connecter au cluster Amazon Redshift. Dans cet exemple, le plugin utilisé est destiné à l'authentification du pilote ODBC standard à l'aide d'un nom de connexion et d'un mot de passe.

```
select driver_version, os_version, plugin_name from sys_connection_log;

driver_version                          |  os_version                       |
 plugin_name
```

```
----------------------------------------+-----------------------------------
+--------------------
Amazon Redshift ODBC Driver 1.4.15.0001 | Darwin 18.7.0 x86_64               | none
Amazon Redshift ODBC Driver 1.4.15.0001 | Linux 4.15.0-101-generic x86_64    | none
```

L'exemple suivant montre la version du système d'exploitation de la machine cliente, la version du pilote et la version du protocole.

```
select os_version, driver_version, protocol_version from sys_connection_log;

os_version                     |  driver_version              | protocol_version
-------------------------------+------------------------------+--------------------
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
```

SYS_COPY_JOB

Utilisez SYS_COPY_JOB pour afficher les détails des commandes COPY JOB.

Cette vue contient les commandes COPY JOB qui ont été créées.

SVV_COPY_JOB est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
job_id	bigint	Identifiant de la tâche de copie.
job_name	character(128)	Nom de la tâche de copie.
iam_role	character(128)	Rôle IAM spécifié dans l'instruction COPY.
job_text	character(256)	Paramètres de l'instruction COPY.

Nom de la colonne	Type de données	Description
is_auto	entier	Indique si COPY JOB est automatiquement exécuté par Amazon Redshift. Un 1 indique vrai, 0 indique faux.
on_error_suspend	entier	Information à utilisation interne uniquement.

SYS_COPY_JOB_DETAIL

Utilisez SYS_COPY_JOB_DETAIL pour afficher les détails des commandes COPY JOB.

Cette vue contient les commandes COPY JOB qui ont été créées. Si COPY JOB tente de charger un fichier mais qu'il échoue, le fichier est ignoré lors des futures tentatives de copie automatique.

SYS_COPY_JOB_DETAIL est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identifiant de l'utilisateur propriétaire de la tâche.
database_name	text	Nom de la base de données contenant le travail.
job_id	entier	Identifiant de la tâche de copie.
job_name	text	Nom de la tâche de copie.
file_location	text	Le nom du compartiment Amazon S3 pour l'entrée.

Nom de la colonne	Type de données	Description
file_name	text	Le nom de clé Amazon S3 pour l'entrée.
file_size	bigint	Taille de l'entrée en octets.
fichier_etag	text	Le tag d'entité Amazon S3 (ETag) pour l'entrée.
job_text	character(256)	Paramètres de l’instruction COPY.
heure_de modification	timestamp	Heure (UTC) à laquelle l'objet S3 a été créé ou modifié lors de l'ingestion. En mode de liste S3, il s'agit de l'heure de modification de l'objet. Pour le mode notification, il s'agit de l'heure à laquelle l'événement de notification s'est produit.
heure de la file d'attente	timestamp	Heure (UTC) à laquelle l'entrée a été ajoutée à la file d'ingestion.
status	char(1)	État de l'entrée. Valeurs valides : `P` pour une ingestion en attente, `I` pour une ingestion ingérée, `E` pour une ingestion erronée et `U` pour une valeur inconnue (indiquant une sorte d'état inattendu).

L'exemple suivant renvoie une ligne pour les entrées ingérées.

```
SELECT * FROM SYS_COPY_JOB_DETAIL WHERE status ilike '%ingested%' limit 1;
```

```
user_id | 100
database_name | dev
job_name | many_job_4_3
job_id | 110702
file_location | saral-sqs-system4623202051-0
file_name | frenzy-9/4623202051/file_0_107
file_size | 11302
file_etag | 51b2d78ac5b5aecf4ee6f8374815ad19
modification_time | 2024-07-15 20:43:14
enqueue_time | 2024-07-15 20:44:24
status | Ingested
```

SYS_COPY_JOB_INFO

Utilisez SYS_COPY_JOB_INFO pour afficher les messages enregistrés à propos d'un COPY JOB.

Cette vue contient des informations sur les erreurs survenues dans un COPY JOB exécuté.

SYS_COPY_JOB_INFO est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
job_id	bigint	Identifiant de la tâche de copie.
job_name	character(128)	Nom de la tâche de copie.
database_name	character(128)	Nom de la base de données.
record_time	timestamp	Heure (UTC) à laquelle le message a été enregistré.
message	personnage (512)	Ce message de l'événement enregistré pour un COPY JOB.

SYS_COPY_REPLACEMENTS

Affiche un journal qui enregistre quand les caractères UTF-8 non valides ont été remplacés par la commande COPY avec l'option ACCEPTINVCHARS. Une entrée de journal est ajoutée à SYS_COPY_REPLACEMENTS pour chacune des 100 premières lignes de chaque tranche de nœud qui nécessitait au moins un remplacement.

Vous pouvez vous servir de cette vue pour examiner les informations sur les groupes de travail sans serveur et les clusters provisionnés.

SYS_COPY_REPLACEMENTS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	ID de l'utilisateur qui a généré la requête.
query_id	bigint	ID de requête. Colonne utilisée pour joindre d'autres tables et vues système.
table_id	entier	ID de la table.
file_name	character (256)	Chemin complet du fichier d'entrée pour la commande COPY.
column_name	character (127)	Premier champ contenant un caractère UTF-8 non valide.
line_number	bigint	Numéro de ligne dans le fichier de données d'entrée qui contient un caractère UTF-8 non valide. -1 indique que le numéro de ligne n'est pas disponible, ce qui est notamment le cas lorsque la copie s'effectue à partir d'un fichier de données en colonnes.

Nom de la colonne	Type de données	Description
raw_line	character (1024)	Données de chargement brutes contenant un caractère UTF-8 non valide.

Exemples de requêtes

L'exemple suivant renvoie les remplacements pour l'opération COPY la plus récente.

```
select query_idp, table_id, file_name, line_number, colname
from sys_copy_replacements
where query = pg_last_copy_id();


 query_id | table_id |   file_name                                                |
 line_number | column_name
 ---------+----------+------------------------------------------------------------
+-------------+--------
    96    |    26    | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt             |
 123 | city
    96    |    26    | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt             |
 456 | city
    96    |    26    | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt             |
 789 | city
    96    |    26    | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt             |
 012 | city
    96    |    26    | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt             |
 119 | city
...
```

SYS_DATASHARE_CHANGE_LOG

Enregistre la vue consolidée pour le suivi des modifications apportées aux unités de partage des données sur les clusters producteur et consommateur.

SYS_DATASHARE_CHANGE_LOG est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	L'ID de l'utilisateur qui effectue l'action.
user_name	varchar(128)	Le nom de l'utilisateur qui effectue l'action.
session_id	entier	L'ID de la séance.
transaction_id	bigint	ID de la transaction.
share_id	entier	L'ID de l'unité de partage des données affecté.
share_name	varchar(128)	Le nom du datashare.
source_database_id	entier	L'ID de la base de données à laquelle appartient l'unité de partage des données.
source_database_name	varchar(128)	Le nom de la base de données à laquelle appartient le datashare.
consumer_database_id	entier	L'ID de la base de données importée depuis l'unité de partage des données.
consumer_database_name	varchar(128)	Le nom de la base de données importée depuis le datashare.
arn	varchar(192)	L'ARN de la ressource qui soutient la base de données importée.
record_time	timestamp	Horodatage de l'action.

Nom de la colonne	Type de données	Description
action	varchar(128)	L'action en cours d'exécution. Les valeurs possibles sont CREATE DATASHARE, DROP DATASHARE, GRANT ALTER, REVOKE ALTER, GRANT SHARE, REVOKE SHARE, ALTER ADD, ALTER REMOVE, ALTER SET, GRANT USAGE, REVOKE USAGE, CREATE DATABASE, GRANT ou REVOKE USAGE sur une base de données partagée, DROP SHARED DATABASE, ALTER SHARED DATABASE.
status	entier	Statut de l'action. Les valeurs possibles sont SUCCESS et ERROR-ERROR CODE.
share_object_type	varchar(64)	Le type d'objet de base de données qui a été ajouté ou supprimé de l'unité de partage des données. Les valeurs possibles sont les schémas, les tables, les colonnes, les fonctions et les vues. Il s'agit d'un champ pour le cluster producteur.
share_object_id	entier	L'ID de l'objet de base de données qui a été ajouté ou supprimé de l'unité de partage des données. Il s'agit d'un champ pour le cluster producteur.
share_object_name	varchar(128)	Le nom de l'objet de base de données qui a été ajouté ou supprimé de l'unité de partage des données. Il s'agit d'un champ pour le cluster producteur.
target_user_type	varchar(16)	Le type d'utilisateurs ou de groupes auxquels un privilège a été accordé. Il s'agit d'un domaine à la fois pour le cluster producteur et consommateur.
target_user_id	entier	L'ID des utilisateurs ou des groupes auxquels un privilège a été accordé. Il s'agit d'un domaine à la fois pour le cluster producteur et consommateur.
target_user_name	varchar(128)	Le nom des utilisateurs ou des groupes auxquels un privilège a été accordé. Il s'agit d'un domaine à la fois pour le cluster producteur et consommateur.

Nom de la colonne	Type de données	Description
consumer_account	varchar(16)	L'ID de compte du consommateur de données. Il s'agit d'un champ pour le cluster producteur.
consumer_namespace	varchar(64)	L'espace de noms du compte consommateur de données. Il s'agit d'un champ pour le cluster producteur.
producer_account	varchar(16)	L'ID du compte producteur auquel appartient l'unité de partage des données. Il s'agit d'un champ pour le cluster consommateur.
producer_namespace	varchar(64)	L'espace de noms du compte producteur auquel appartient l'unité de partage des données. Il s'agit d'un champ pour le cluster consommateur.
attribute_name	varchar(64)	Le nom d'un attribut de l'unité de partage des données ou de la base de données partagée.
attribute_value	varchar(128)	La valeur d'un attribut de l'unité de partage des données ou de la base de données partagée.
message	varchar(512)	Le message d'erreur lorsqu'une action échoue.

Exemples de requêtes

L'exemple suivant montre une vue SYS_DATASHARE_CHANGE_LOG.

```
SELECT DISTINCT action
FROM sys_datashare_change_log
WHERE share_object_name LIKE 'tickit%';

         action
 ----------------------
  "ALTER DATASHARE ADD"
```

SYS_DATASHARE_CROSS_REGION_USAGE

Utilisez la vue SYS_DATASHARE_CROSS_REGION_USAGE pour obtenir un résumé de l'utilisation des données transférées d'une région à l'autre, causée par une requête d'unité de partage des données entre régions. SYS_DATASHARE_CROSS_REGION_USAGE agrège les détails au niveau du segment.

SYS_DATASHARE_CROSS_REGION_USAGE n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
query_id	entier	ID de la requête. Utilisez cette valeur pour joindre d'autres tables et vues du système.
child_query_séquence	entier	La séquence de la requête utilisateur réécrite, en commençant par 1.
segment_id	bigint	Numéro du segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes.
start_time	time	Heure UTC à laquelle le transfert de données a commencé.
end_time	time	Heure UTC à laquelle le transfert de données s'est terminé.
transferred_data	bigint	Nombre d'octets de données transférés d'une région productrice à une région consommatrice.
source_region	char(255)	La région productrice à partir de laquelle la requête a transféré des données.

Exemples de requêtes

L'exemple suivant montre une vue SYS_DATASHARE_CROSS_REGION_USAGE.

```
SELECT query_id, segment_id, transferred_data, source_region
```

```
from sys_datashare_cross_region_usage
where query_id = pg_last_query_id()
order by query_id, segment_id;

  query_id | segment_id | transferred_data | source_region
-----------+------------+------------------+---------------
    200048 |          2 |          4194304 |    us-west-1
    200048 |          2 |          4194304 |    us-east-2
```

SYS_DATASHARE_USAGE_CONSUMER

Enregistre l'activité et l'utilisation des datashares. Cette vue n’est pertinente que pour le cluster consommateur.

SYS_DATASHARE_USAGE_CONSUMER est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	L’ID de l’utilisateur à l’origine de la requête.
session_id	entier	ID du processus de leader exécutant la requête.
transaction_id	bigint	Le contexte de la transaction actuelle.
request_id	varchar(50)	L’ID unique de l’appel d’API demandé.
request_type	varchar(25)	Le type de la requête envoyée au cluster producteur.
transaction_uid	varchar(50)	ID unique de la transaction.

Nom de la colonne	Type de données	Description
record_time	timestamp	L'heure à laquelle l'action est enregistrée.
status	entier	Le statut de l'appel d'API demandé.
error_message	varchar(512)	Un message d'erreur.

Exemples de requêtes

L'exemple suivant montre la vue SYS_DATASHARE_USAGE_CONSUMER.

```
SELECT request_type, status, trim(error) AS error
FROM sys_datashare_usage_consumer

  request_type  | status | error_message
----------------+--------+---------------
 "GET RELATION" |   0    |
```

SYS_DATASHARE_USAGE_PRODUCER

Enregistre l'activité et l'utilisation des datashares. Cette vue n'est pertinente que sur le cluster producteur.

SYS_DATASHARE_USAGE_PRODUCER est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
share_id	entier	L'ID d'objet (OID) de l'unité de partage des données.

Nom de la colonne	Type de données	Description
share_name	varchar(128)	Le nom du datashare.
request_id	varchar(50)	L'ID unique de l'appel d'API demandé.
request_type	varchar(25)	Le type de la requête envoyée au cluster producteur.
object_type	varchar(64)	Le type de l'objet partagé à partir de l'unité de partage des données. Les valeurs possibles sont les schémas, les tables, les colonnes, les fonctions et les vues.
object_oid	entier	L'ID de l'objet partagé à partir de l'unité de partage des données.
nom d'objet	varchar(128)	Le nom de l'objet partagé à partir de l'unité de partage des données.
consumer_account	varchar(16)	Le compte consommateur relié au datashare pour le partage.
consumer_namespace	varchar(64)	L'espace de noms du compte consommateur relié à l'unité de partage des données pour le partage.
consumer_transaction_uid	varchar(50)	L'ID de transaction unique de l'instruction sur le cluster consommateur.
record_time	timestamp	L'heure à laquelle l'action est enregistrée.
status	entier	Le statut du datashare.
error_message	varchar(512)	Un message d'erreur.
consumer_region	char(64)	Région dans laquelle est situé le cluster consommateur.

Exemples de requêtes

L'exemple suivant montre la vue SYS_DATASHARE_USAGE_PRODUCER.

```
SELECT DISTINCT
FROM sys_datashare_usage_producer
WHERE object_name LIKE 'tickit%';

   request_type
 ------------------
   "GET RELATION"
```

SYS_EXTERNAL_QUERY_DETAIL

Utilisez SYS_EXTERNAL_QUERY_DETAIL pour afficher les détails des requêtes au niveau d'un segment. Chaque ligne représente un segment d'une requête WLM particulière avec des détails tels que le nombre de lignes traitées, le nombre d'octets traités et les informations de partition des tables externes dans Amazon S3. Chaque ligne de cette vue aura également une entrée correspondante dans la vue SYS_QUERY_DETAIL, sauf que cette vue contient des informations plus détaillées relatives au traitement des requêtes externes.

SYS_EXTERNAL_QUERY_DETAIL est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identificateur de l'utilisateur qui a envoyé la requête.
query_id	bigint	Identifiant de requête de la requête externe.
transaction_id	bigint	Identificateur de transaction.
child_query_séquence	entier	La séquence de la requête utilisateur réécrite. Commence par 0, similaire à segment_id.

Nom de la colonne	Type de données	Description
segment_id	entier	Identificateur du segment de la requête.
source_type	character(32)	Type de source de données de la requête, il peut s'agir de S3 pour Redshift Spectrum et de PG pour une requête fédérée.
start_time	timestamp	Heure à laquelle la requête a commencé.
end_time	timestamp	Heure à laquelle la requête s'est terminée.
duration	bigint	Temps (microsecondes) passé sur la requête.
total_partitions	entier	Nombre de partitions requises par une requête Amazon S3.
qualified_partitions	entier	Nombre de partitions analysées par une requête Amazon S3.
scanned_files	bigint	Nombre de fichiers Amazon S3 analysés.
returned_rows	bigint	Nombre de lignes analysées pour une requête Amazon S3 ou nombre de lignes renvoyées pour une requête fédérée.

Nom de la colonne	Type de données	Description
returned_bytes	bigint	Nombre d'octets analysés pour une requête Amazon S3 ou nombre d'octets renvoyés pour une requête fédérée.
file_format	text	Format de fichier des fichiers Amazon S3.
file_location	text	Emplacement Amazon S3 de la table externe.
external_query_text	text	Texte de requête au niveau du segment pour une requête fédérée.
warning_message	character(4000)	Le message d'avertissement affiché lors de l'exécution de la requête.
table_name	character(136)	Nom de la table de l'étape en cours d'opération.
is_recursive	character(1)	Si l'analyse des sous-dossiers est récursive.
is_nested	character(1)	Indique si le type de données de la colonne imbriquée est accessible.
s3list_time	bigint	Durée de la liste des fichiers en millisecondes.
get_partition_time	long	Temps passé à répertorier et à qualifier les partitions pour un objet externe donné depuis Apache Hive. AWS Glue Data Catalog

Exemples de requêtes

La requête suivante montre les détails de la requête externe.

```
SELECT query_id,
       segment_id,
       start_time,
       end_time,
       total_partitions,
       qualified_partitions,
       scanned_files,
       returned_rows,
       returned_bytes,
       trim(external_query_text) query_text,
       trim(file_location) file_location
FROM sys_external_query_detail
ORDER BY query_id, start_time DESC
LIMIT 2;
```

Exemple de sortie.

```
 query_id | segment_id |         start_time         |          end_time
   | total_partitions | qualified_partitions | scanned_files | returned_rows |
 returned_bytes | query_text | file_location
----------+------------+----------------------------+----------------------------
+------------------+----------------------+--------------+--------------
+----------------+------------+---------------
   763251 |          0 | 2022-02-15 22:32:23.312448 | 2022-02-15 22:32:24.036023 |
          3 |                    3 |             3 |         38203 |        2683414 |
          |
   763254 |          0 | 2022-02-15 22:32:40.17103  | 2022-02-15 22:32:40.839313 |
          3 |                    3 |             3 |         38203 |        2683414 |
          |
```

SYS_EXTERNAL_QUERY_ERROR

Vous pouvez interroger la vue système SYS_EXTERNAL_QUERY_ERROR pour obtenir des informations sur les erreurs d'analyse Redshift Spectrum. SYS_EXTERNAL_QUERY_ERROR affiche un exemple d'erreurs journalisées. La valeur par défaut est de 10 entrées par requête.

SYS_EXTERNAL_QUERY_ERROR est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs

propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identifiant de l'utilisateur qui a généré cette ligne.
query_id	bigint	Identifiant de la requête qui a généré cette ligne.
file_location	char(256)	L'emplacement des données à interroger.
rowid	char(2100)	Emplacement de l'erreur dans le fichier. Les parties `rowid` sont séparées par `:` (deux-points), et des parties supplémentaires peuvent être ajoutées plus tard. `row_offset :row_group :row_id` Un row_offset correspond au décalage (en octets) de la ligne dans le fichier et est défini sur `-1` pour les formats de fichier non pris en charge. Une table est divisée en row_groups, et chaque groupe possède des lignes avec des row_ids distincts.
column_name	char(127)	Le nom de la colonne renvoyée par la requête.
original_value	char(1024)	Valeur d'origine interrogée.
modified_value	char(1024)	Valeur modifiée renvoyée en fonction de l'option de configuration de gestion des données spécifiée dans la requête.
déclencheur	char(128)	Option de gestion des données spécifiée dans la requête.

Nom de la colonne	Type de données	Description
action	char(128)	Action associée à l'option de gestion des données spécifiée dans la requête.
action_value	char(128)	Paramètre de valeur d'action associé à l'option de traitement des données spécifiée dans la requête.
error_code	entier	Code de résultat de l'option de traitement des données spécifiée dans la requête.

Exemple de requête

La requête suivante renvoie la liste des lignes pour lesquelles des opérations de traitement des données ont été effectuées.

```
SELECT * FROM sys_external_query_error;
```

La requête renvoie des résultats semblables à ce qui suit.

```
  user_id   query_id  file_location                                rowid
column_name           original_value             modified_value       trigger
action               action_value                 error_code
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0
league_name           Barclays Premier League    Barclays Premier Lea UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0
league_nspi           34595                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:1
league_nspi           34151                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2
league_name           Barclays Premier League    Barclays Premier Lea UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2
league_nspi           33223                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
```

```
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3
league_name           Barclays Premier League    Barclays Premier Lea UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3
league_nspi           32808                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:4
league_nspi           32790                      32767                UNSPECIFIED
OVERFLOW_VALUE                                    199
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:5
league_name           Spanish Primera Division   Spanish Primera Divi UNSPECIFIED
TRUNCATE                                          156
    100     1574007   s3://spectrum-uddh/league/spi_global_rankings.0:6
league_name           Spanish Primera Division   Spanish Primera Divi UNSPECIFIED
TRUNCATE                                          156
```

SYS_INTEGRATION_ACTIVITY

SYS_INTEGRATION_ACTIVITY affiche les détails des cycles d'intégration terminés.

SYS_INTEGRATION_ACTIVITY n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Pour en savoir plus sur les intégrations zéro ETL, consultez Utilisation des intégrations zéro ETL du Guide de gestion Amazon Redshift.

Colonnes de la table

Nom de la colonne	Type de données	Description
integration_id	character (128)	Identifiant associé à l'intégration.
target_database	character (128)	Base de données d'Amazon Redshift qui reçoit les données d'intégration.
source	character (128)	Données source de l'intégration. Les types possibles incluent `MySQL` et `PostgreSQL` .

Nom de la colonne	Type de données	Description
checkpoint_name	character (128)	Nom du point de contrôle qui réplique les coordonnées binlog.
checkpoint_type	character (16)	Type de point de contrôle. Les valeurs admises incluent : `snapshot`, `cdc`.
checkpoint_bytes	bigint	Nombre d'octets de ce point de contrôle.
last_commit_timestamp	timestamp	Horodatage de la dernière validation à ce point de contrôle.
modified_tables	entier	Nombre de tables modifiées dans le point de contrôle.
integration_start_time	time	Heure (UTC) à laquelle l'intégration a commencé pour ce point de contrôle.
integration_end_time	time	Heure (UTC) à laquelle l'intégration s'est terminée pour ce point de contrôle.

Exemples de requêtes

La commande SQL suivante affiche le journal des intégrations.

```
select * from sys_integration_activity;

          integration_id              | target_database | source |
   checkpoint_name                  | checkpoint_type  | checkpoint_bytes |
 last_commit_timestamp   | modified_tables |   integration_start_time   |
 integration_end_time
-------------------------------------+-----------------+--------
+--------------------------------------------+------------------+------------------
+-------------------------+-----------------+----------------------------
+----------------------------
 76b15917-afae-4447-b7fd-08e2a5acce7b |   demo1         | MySQL  | checkpoints/
checkpoint_3_241_3_510.json     |        cdc       |        762       | 2023-05-10
```

```
 23:00:14.201 |         1       | 2023-05-10 23:00:45.054265 | 2023-05-10
 23:00:46.339826
 76b15917-afae-4447-b7fd-08e2a5acce7b |   demo1         | MySQL  | checkpoints/
checkpoint_3_16329_3_17839.json |        cdc       |       13488      | 2023-05-11
 01:33:57.411 |         2       | 2023-05-11 02:19:09.440121 | 2023-05-11
 02:19:16.090492
 76b15917-afae-4447-b7fd-08e2a5acce7b |   demo1         | MySQL  | checkpoints/
checkpoint_3_5103_3_5532.json   |        cdc       |        1657      | 2023-05-10
 23:13:14.205 |         2       | 2023-05-10 23:13:23.545487 | 2023-05-10
 23:13:25.652144
```

SYS_INTEGRATION_TABLE_ACTIVITY

SYS_INTEGRATION_TABLE_ACTIVITY affiche les détails des activités d'insertion, de suppression et de mise à jour des intégrations sans ETL. Une ligne est ajoutée pour chaque ingestion terminée.

Un super-utilisateur peut afficher toutes les lignes de cette table.

Pour plus d'informations, consultez la section Intégrations Zero-ETL.

Colonnes de la table

Nom de la colonne	Type de données	Description
integration_id	character (128)	Identifiant associé à l’intégration.
checkpoint_name	character (128)	Le nom du point de contrôle.
target_database	character (128)	Nom de la base de données Amazon Redshift.
nom_schéma	character (128)	Nom du schéma Amazon Redshift.
table_name	character (128)	Nom de la table.
table_id	entier	Identifiant de la table.

Nom de la colonne	Type de données	Description
record_time	timestamp	Heure (UTC) à laquelle cette modification s'est terminée.
transaction_id	bigint	Identificateur de transaction.
lignes_insérées	bigint	Nombre de lignes insérées lors de l'ingestion.
lignes_supprimées	bigint	Nombre de lignes supprimées par l'ingestion.
rangées mises à jour	bigint	Le nombre de lignes mises à jour par l'ingestion.
octets_ingested	bigint	Le nombre d'octets ingérés.

Exemples de requêtes

La commande SQL suivante affiche l'activité de l'intégration.

```
select * from sys_integration_table_activity;

          integration_id              | checkpoint_name | target_database | schema_name
 |     table_name    | table_id     | record_time                | transaction_id  |
 inserted_rows  | deleted_rows | updated_rows | bytes_ingested
---------------------------------------+-----------------+-----------------
+-------------+-------------------+-------------+---------------------------
+-----------------+----------------+-------------+-------------+---------------
 4798e675-8f9f-4686-b05f-92c538e19629 |                 | sample_test2    |    sample
 | SampleTestChannel |  111276      | 2023-05-12 12:40:30.656625 | 7736            |  2
             | 0            | 0            | 125
```

SYS_INTEGRATION_TABLE_STATE_CHANGE

SYS_INTEGRATION_TABLE_STATE_CHANGE affiche des détails sur les journaux de modification de l'état des tables pour les intégrations.

Un super-utilisateur peut afficher toutes les lignes de cette table.

Pour plus d'informations, consultez la section Utilisation des intégrations sans ETL.

Colonnes de la table

Nom de la colonne	Type de données	Description
integration_id	character (128)	Identifiant associé à l'intégration.
database_name	character (128)	Nom de la base de données Amazon Redshift.
nom_schéma	character (128)	Nom du schéma Amazon Redshift.
table_name	character (128)	Nom de la table.
nouvel_État	character (128)	État de la table. Les valeurs possibles sont `Synced`, `ResyncRequired` , `ResyncInitiated` , `Deleted`, `Failed` et `ResyncDeleted` .
table_last_replicated_checkpoint	character (128)	Coordonnées du journal synchronisé actuel.
Motif du changement d'état	character (256)	Motif de la dernière transition d'état.
record_time	timestamp	Heure (UTC) à laquelle cet enregistrement a été mis à jour.

Exemples de requêtes

La commande SQL suivante affiche le journal des intégrations.

```
select * from sys_integration_table_state_change;

           integration_id            | database_name | schema_name | table_name
 | new_state |  table_last_replicated_checkpoint   | state_change_reason |
 record_time
```

```
--------------------------------------+--------------+------------+------------
+----------+-------------------------------------+--------------------
+---------------------------
 99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb       | sbtest80t3s | sbtest79   |
 Synced    | {"txn_seq":9834,"txn_id":126597515} |                   | 2023-09-20
 19:39:50.087868
 99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb       | sbtest80t3s | sbtest56   |
 Synced    | {"txn_seq":9834,"txn_id":126597515} |                   | 2023-09-20
 19:39:45.54005
 99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb       | sbtest80t3s | sbtest50   |
 Synced    | {"txn_seq":9834,"txn_id":126597515} |                   | 2023-09-20
 19:40:20.362504
 99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb       | sbtest80t3s | sbtest18   |
 Synced    | {"txn_seq":9834,"txn_id":126597515} |                   | 2023-09-20
 19:40:32.544084
 99108e72-1cfd-414f-8cc0-0216acefac77 | perfdb       | sbtest40t3s | sbtest23   |
 Synced    | {"txn_seq":9834,"txn_id":126597515} |                   | 2023-09-20
 15:49:05.186209
```

SYS_LOAD_DETAIL

renvoie les informations pour suivre ou dépanner une charge de données.

Cette vue enregistre la progression de chaque fichier de données lorsqu'il est chargé dans une table de base de données.

Cette vue est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	ID de l'utilisateur qui a généré l'entrée.
query_id	entier	ID de requête.
file_name	character(256)	Nom du fichier à charger.

Nom de la colonne	Type de données	Description
bytes_scanned	entier	Le nombre d'octets analysés à partir du fichier dans Amazon S3.
lines_scanned	entier	Nombre de lignes du fichier de charge analysées. Ce nombre peut ne pas correspondre au nombre de lignes qui sont effectivement chargées. Par exemple, la charge peut analyser, mais tolérer, un certain nombre d'enregistrements incorrects, en fonction de l'option MAXERROR de la commande COPY.
record_time	timestamp	Heure à laquelle cette entrée a été mise à jour pour la dernière fois.
splits_scanned	Nombre de divisions de ce fichier.	Nombre de divisions de ce fichier.
start_time	timestamp	Heure à laquelle le traitement du fichier a commencé.
end_time	timestamp	Heure à laquelle le traitement du fichier s'est terminé.

Exemples de requêtes

L'exemple suivant renvoie les détails de la dernière opération COPY.

```
select query_id, trim(file_name) as file, record_time
from sys_load_detail
where query_id = pg_last_copy_id();

 query_id |               file                |          record_time
----------+-----------------------------------+----------------------------
 28554    | s3://dw-tickit/category_pipe.txt | 2013-11-01 17:14:52.648486
(1 row)
```

La requête suivante contient les entrées d'un nouveau chargement de tables dans la base de données TICKIT :

```
select query_id, trim(file_name), record_time
from sys_load_detail
where file_name like '%tickit%' order by query_id;

 query_id |           btrim           |          record_time
----------+---------------------------+---------------------------
 22475    | tickit/allusers_pipe.txt | 2013-02-08 20:58:23.274186
 22478    | tickit/venue_pipe.txt    | 2013-02-08 20:58:25.070604
 22480    | tickit/category_pipe.txt | 2013-02-08 20:58:27.333472
 22482    | tickit/date2008_pipe.txt | 2013-02-08 20:58:28.608305
 22485    | tickit/allevents_pipe.txt| 2013-02-08 20:58:29.99489
 22487    | tickit/listings_pipe.txt | 2013-02-08 20:58:37.632939
 22593    | tickit/allusers_pipe.txt | 2013-02-08 21:04:08.400491
 22596    | tickit/venue_pipe.txt    | 2013-02-08 21:04:10.056055
 22598    | tickit/category_pipe.txt | 2013-02-08 21:04:11.465049
 22600    | tickit/date2008_pipe.txt | 2013-02-08 21:04:12.461502
 22603    | tickit/allevents_pipe.txt| 2013-02-08 21:04:14.785124
 22605    | tickit/listings_pipe.txt | 2013-02-08 21:04:20.170594

(12 rows)
```

Le fait qu'un enregistrement soit écrit dans le fichier journal pour cette vue système ne signifie pas que la charge a été validée avec succès dans le cadre de la transaction contenante. Pour vérifier les validations de charge, interrogez la vue STL_UTILITYTEXT et recherchez l'enregistrement COMMIT qui correspond à une transaction COPY. Par exemple, cette requête joint SYS_LOAD_DETAIL et STL_QUERY à partir d'une sous-requête sur STL_UTILITYTEXT :

```
select l.query_id,rtrim(l.file_name),q.transaction_id
from sys_load_detail l, sys_query_text q
where l.query_id=q.query_id
and exists
(select xid from stl_utilitytext where xid=q.transaction_id and
 rtrim("text")='COMMIT');

 query_id |           rtrim           |  transaction_id
----------+---------------------------+-----------------
 22600    | tickit/date2008_pipe.txt  | 68311
 22480    | tickit/category_pipe.txt  | 68066
  7508    | allusers_pipe.txt         | 23365
  7552    | category_pipe.txt         | 23415
  7576    | allevents_pipe.txt        | 23429
  7516    | venue_pipe.txt            | 23390
```

```
  7604    | listings_pipe.txt         | 23445
 22596    | tickit/venue_pipe.txt     | 68309
 22605    | tickit/listings_pipe.txt  | 68316
 22593    | tickit/allusers_pipe.txt  | 68305
 22485    | tickit/allevents_pipe.txt | 68071
  7561    | allevents_pipe.txt        | 23429
  7541    | category_pipe.txt         | 23415
  7558    | date2008_pipe.txt         | 23428
 22478    | tickit/venue_pipe.txt     | 68065
   526    | date2008_pipe.txt         |  2572
  7466    | allusers_pipe.txt         | 23365
 22482    | tickit/date2008_pipe.txt  | 68067
 22598    | tickit/category_pipe.txt  | 68310
 22603    | tickit/allevents_pipe.txt | 68315
 22475    | tickit/allusers_pipe.txt  | 68061
   547    | date2008_pipe.txt         |  2572
 22487    | tickit/listings_pipe.txt  | 68072
  7531    | venue_pipe.txt            | 23390
  7583    | listings_pipe.txt         | 23445
(25 rows)
```

SYS_LOAD_ERROR_DETAIL

Utilisez SYS_LOAD_ERROR_DETAIL pour afficher les détails des erreurs de commande COPY. Chaque ligne représente une commande COPY. Elle contient les commandes COPY en cours d'exécution et terminées.

SYS_LOAD_ERROR_DETAIL est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identifiant de l'utilisateur qui a envoyé la copie.
query_id	bigint	Identifiant de la requête de la copie.

Nom de la colonne	Type de données	Description
transaction_id	bigint	Identificateur de transaction.
session_id	entier	Identifiant de processus du processus exécutant la copie.
database_name	character(64)	Nom de la base de données à laquelle l'utilisateur était connecté lorsque la copie a été émise.
table_id	entier	L'identificateur de table.
start_time	timestamp	Heure (UTC) au début de la copie.
file_name	character(256)	Chemin d'accès complet au fichier d'entrée à charger.
line_number	bigint	Numéro de ligne dans le fichier de chargement avec l'erreur. Lorsque vous chargez un fichier JSON, le numéro de ligne de la dernière ligne de l'objet JSON avec l'erreur.
column_name	character(127)	Champ avec l'erreur.
type_colonne	character(10)	Type de données du champ avec l'erreur.

Nom de la colonne	Type de données	Description
column_length	character(10)	Longueur de la colonne, le cas échéant. Ce champ est rempli lorsque le type de données a une longueur limite. Par exemple, pour une colonne avec le type de données « character(3) », cette colonne contient la valeur « 3 ».
position	entier	Position de l'erreur dans le champ.
error_code	entier	Code de l'erreur.
error_message	character(512)	Explication de l'erreur.

Exemples de requêtes

La requête suivante affiche les détails des erreurs de chargement de la commande COPY pour une requête spécifique.

```
SELECT query_id,
       table_id,
       start_time,
       trim(file_name) AS file_name,
       trim(column_name) AS column_name,
       trim(column_type) AS column_type,
       trim(error_message) AS error_message
FROM sys_load_error_detail
WHERE query_id = 762949
ORDER BY start_time
LIMIT 10;
```

Exemple de sortie.

```
 query_id | table_id |         start_time         |               file_name
      | column_name | column_type |                 error_message
```

```
----------+----------+--------------------------
+----------------------------------------+------------+------------
+-----------------------------------------------
   762949 |   137885 | 2022-02-15 22:14:46.759151 | s3://load-test/copyfail/
wrong_format_000 | id          | int4        | Invalid digit, Value 'a', Pos 0, Type:
 Integer
   762949 |   137885 | 2022-02-15 22:14:46.759151 | s3://load-test/copyfail/
wrong_format_001 | id          | int4        | Invalid digit, Value 'a', Pos 0, Type:
 Integer
```

SYS_LOAD_HISTORY

Utilisez SYS_LOAD_HISTORY pour afficher les détails des commandes COPY. Chaque ligne représente une commande COPY avec des statistiques accumulées pour certains champs. Elle contient les commandes COPY en cours d'exécution et terminées.

SYS_LOAD_HISTORY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identifiant de l'utilisateur qui a envoyé la copie.
query_id	bigint	Identifiant de la requête de la copie.
transaction_id	bigint	Identificateur de transaction.
session_id	entier	Identifiant de processus du processus exécutant la copie.
database_name	text	Nom de la base de données à laquelle l'utilisateur était connecté lorsque l'opération a été émise.

Nom de la colonne	Type de données	Description
État	text	Statut de la tâche. Les valeurs valides sont `running`, `completed` , `aborted`.
table_name	text	Nom de la table dans laquelle la table est copiée.
start_time	timestamp	Heure à laquelle la copie a commencé.
end_time	timestamp	Heure à laquelle la copie s'est terminée.
duration	bigint	Durée (en microsecondes) passée dans la commande COPY.
data_source	text	Emplacement Amazon S3 des fichiers entrés à copier.
file_format	text	Le format du fichier source. Les formats incluent csv, txt, json, avro, orc ou parquet.
loaded_rows	bigint	Nombre total de lignes copiées dans une table.
loaded_bytes	bigint	Nombre d'octets copiés dans une table.
source_file_count	entier	Nombre de fichiers comptabilisés dans les fichiers source.
source_file_bytes	bigint	Nombre d'octets dans les fichiers source.

Nom de la colonne	Type de données	Description
file_count_scanned	entier	Nombre de fichiers analysés depuis Amazon S3.
file_bytes_scanned	bigint	Le nombre d'octets analysés à partir du fichier dans Amazon S3.
error_count	bigint	Nombre d'erreurs comptabilisées.
copy_job_id	bigint	Identifiant de la tâche de copie. Un 0 indique qu'il n'y a aucun identifiant de tâche.

Exemples de requêtes

La requête suivante affiche les lignes, les octets, les tables et la source de données chargés des commandes COPY spécifiques.

```
SELECT query_id,
       table_name,
       data_source,
       loaded_rows,
       loaded_bytes
FROM sys_load_history
WHERE query_id IN (6389,490791,441663,74374,72297)
ORDER BY query_id,
         data_source DESC;
```

Exemple de sortie.

```
 query_id |    table_name    |                                data_source
              | loaded_rows | loaded_bytes
----------+------------------
+-------------------------------------------------------------------------+-------------
+---------------
     6389 | store_returns    | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
store_returns/    |   287999764 | 1196240296158
```

```
    72297 | web_site         | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
web_site/         |          54 |         43808
    74374 | ship_mode        | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
ship_mode/        |          20 |          1320
   441663 | income_band      | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
income_band/      |          20 |          2152
   490791 | customer_address | s3://load-test/data-sources/tpcds/2.8.0/textfile/1T/
customer_address/ |     6000000 |     722924305
```

La requête suivante affiche les lignes, les octets, les tables et la source de données chargés des commandes COPY.

```
SELECT query_id,
       table_name,
       data_source,
       loaded_rows,
       loaded_bytes
FROM sys_load_history
ORDER BY query_id DESC
LIMIT 10;
```

Exemple de sortie.

```
 query_id |       table_name       |                                data_source
                          | loaded_rows |  loaded_bytes
----------+------------------------
+-------------------------------------------------------------------------------
+-------------+-----------------
   491058 | web_site               | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_site/               |          54 |           43808
   490947 | web_sales              | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_sales/              |   720000376 |  22971988122819
   490923 | web_returns            | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_returns/            |    71997522 |     96597496325
   490918 | web_page               | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/web_page/               |        3000 |            1320
   490907 | warehouse              | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/warehouse/              |          20 |            1320
   490902 | time_dim               | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/time_dim/               |       86400 |            1320
   490876 | store_sales            | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/store_sales/            |  2879987999 | 151666241887933
```

```
   490870 | store_returns          | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/store_returns/          |   287999764 |   1196405607941
   490865 | store                  | s3://load-test/data-sources/tpcds/2.8.0/
textfile/1T/store/                  |        1002 |          365507
```

La requête suivante affiche les lignes et les octets chargés quotidiennement de la commande COPY.

```
SELECT date_trunc('day',start_time) AS exec_day,
       SUM(loaded_rows) AS loaded_rows,
       SUM(loaded_bytes) AS loaded_bytes
FROM sys_load_history
GROUP BY exec_day
ORDER BY exec_day DESC;
```

Exemple de sortie.

```
      exec_day       | loaded_rows |   loaded_bytes
---------------------+-------------+------------------
 2022-01-20 00:00:00 |  6347386005 |  258329473070606
 2022-01-19 00:00:00 | 19042158015 |  775198502204572
 2022-01-18 00:00:00 | 38084316030 | 1550294469446883
 2022-01-17 00:00:00 | 25389544020 | 1033271084791724
 2022-01-16 00:00:00 | 19042158015 |  775222736252792
 2022-01-15 00:00:00 | 19834245387 |  798122849155598
 2022-01-14 00:00:00 | 75376544688 | 3077040926571384
```

SYS_MV_REFRESH_HISTORY

Les résultats incluent des informations sur l'historique d'actualisation de toutes les vues matérialisées. Les résultats incluent le type d'actualisation, tel que manuel ou automatique, et le statut de l'actualisation la plus récente.

SYS_MV_REFRESH_HISTORY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identificateur de l'utilisateur qui a soumis à l'actualisation.
session_id	entier	Identificateur du processus exécutant l'actualisation de la vue matérialisée.
transaction_id	bigint	Identificateur de transaction.
database_name	char(128)	Base de données contenant la vue matérialisée.
nom_schéma	char(128)	Schéma de la vue matérialisée.
mv_id	bigint	Identifiant de l'objet de la vue matérialisée.
mv_name	char(128)	Nom de la vue matérialisée.
refresh_type	char(32)	Type d'actualisation, par exemple manuel ou automatique.
État	text	Statut de l'actualisation. Pour des informations détaillées sur les statuts, consultez la colonne de statut pour SVL_MV_REFRESH_STATUS.
start_time	timestamp	Heure de début de l'actualisation.
end_time	timestamp	Heure de fin de l'actualisation.

Nom de la colonne	Type de données	Description
duration	bigint	Durée en microsecondes nécessaire pour actualiser la vue matérialisée.

Exemples de requêtes

La requête suivante montre l'historique d'actualisation des vues matérialisées.

```
SELECT user_id,
     session_id,
     transaction_id,
     database_name,
     schema_name,
     mv_id,
     mv_name,
     refresh_type,
     status,
     start_time,
     end_time,
     duration
     from sys_mv_refresh_history;
```

La requête renvoie l'exemple de sortie suivant :

```
 user_id | session_id | transaction_id | database_name | schema_name                |
 mv_id  |  mv_name           |  refresh_type  |  status
         |  start_time                |  end_time                  |  duration
---------+------------+----------------+--------------
+----------------------------+--------+--------------------+---------------
+-----------------------------------------------+----------------------------
+----------------------------+-----------
       1 | 1073815659 |          15066 | dev           | test_stl_mv_refresh_schema |
 203762 | mv_incremental     | Manual         | MV was already updated
         | 2023-10-26 15:59:20.952179 | 2023-10-26 15:59:20.952866 |      687
       1 | 1073815659 |          15068 | dev           | test_stl_mv_refresh_schema |
 203771 | mv_nonincremental  | Manual         | MV was already updated
         | 2023-10-26 15:59:21.008049 | 2023-10-26 15:59:21.008658 |      609
```

```
      1 | 1073815659 |          15070 | dev           | test_stl_mv_refresh_schema |
 203779 | mv_refresh_error   | Manual         | MV was already updated
        | 2023-10-26 15:59:21.064252 | 2023-10-26 15:59:21.064885 |      633
      1 | 1073815659 |          15074 | dev           | test_stl_mv_refresh_schema
 | 203762 | mv_incremental     | Manual         | Refresh successfully updated MV
 incrementally   | 2023-10-26 15:59:29.693329 | 2023-10-26 15:59:43.482842 | 13789513
      1 | 1073815659 |          15076 | dev           | test_stl_mv_refresh_schema |
 203771 | mv_nonincremental  | Manual         | Refresh successfully recomputed MV from
 scratch | 2023-10-26 15:59:43.550184 | 2023-10-26 15:59:47.880833 |  4330649
      1 | 1073815659 |          15078 | dev           | test_stl_mv_refresh_schema |
 203779 | mv_refresh_error   | Manual         | Refresh failed due to an internal error
        | 2023-10-26 15:59:47.949052 | 2023-10-26 15:59:52.494681 |  4545629
(6 rows)
```

SYS_MV_STATE

Les résultats contiennent des informations sur l'état de toutes les vues matérialisées. Cette vue contient des informations sur la table de base, les propriétés du schéma et des informations sur les événements récents, comme la suppression d'une colonne.

SYS_MV_STATE est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	bigint	ID de l'utilisateur qui a créé l'événement.
transaction_id	bigint	ID de transaction de l'événement.
database_name	char(128)	Base de données contenant la vue matérialisée.
event_desc	char(500)	Événement ayant entraîné le changement d'état. Exemples de valeur possible :

Nom de la colonne	Type de données	Description
		• Le type de colonne a été modifié • La colonne a été supprimée • La colonne a été renommée • Le nom du schéma a été modifié • Conversion de petite table • TRUNCATE • Vacuum Notez qu'il existe d'autres valeurs possibles pour cette colonne.
start_time	timestamp	Heure de début de l'événement.
base_table_database_name	char(128)	Nom de la base de données de la table de base.
base_table_schema	char(128)	Schéma de la table de base.
base_table_name	char(128)	Nom de la table de base.
mv_schema	char(128)	Schéma de la vue matérialisée.
mv_name	char(128)	Nom de la vue matérialisée.
state	character(32)	État modifié de la vue matérialisée, à savoir : • Recompute (Recalculer) • Unrefreshable (Inactualisable)

Exemples de requêtes

La requête suivante affiche l'état de la vue matérialisée.

```
select * from sys_mv_state;
```

La requête renvoie l'exemple de sortie suivant :

```
 user_id | transaction_id | database_name | event_desc                  | start_time
              | base_table_database_name | base_table_schema | base_table_name     |
 mv_schema  | mv_name                    | state
---------+----------------+---------------+-----------------------------
+-----------------------------+--------------------------+------------------
+---------------------+-------------+----------------------------+--------------
 106     | 12720          | tickit_db     | TRUNCATE                    | 2023-07-26
 14:59:12.788268 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_a1f3f862 | Recompute
 106     | 12724          | tickit_db     | ALTER TABLE ALTER DISTSTYLE | 2023-07-26
 14:59:51.409014 | tickit_db                | mv_schema         | test_table_58102435 |
 mv_schema   | materialized_view_ca746631 | Recompute
 106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
 14:59:12.822928 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_5750a8d4 | Unrefreshable
 106     | 12727          | tickit_db     | Table was renamed           | 2023-07-26
 15:00:08.051244 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_5750a8d4 | Unrefreshable
 106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
 14:59:12.857755 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_5750a8d4 | Unrefreshable
 106     | 12727          | tickit_db     | Table was renamed           | 2023-07-26
 15:00:08.051358 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_5ef0d754 | Unrefreshable
 106     | 12720          | tickit_db     | TRUNCATE                    | 2023-07-26
 14:59:12.788159 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_5750a8d4 | Recompute
 106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
 14:59:12.857799 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_a1f3f862 | Unrefreshable
 106     | 12720          | tickit_db     | TRUNCATE                    | 2023-07-26
 14:59:12.788327 | tickit_db                | mv_schema         | test_table_95d6d861 |
 mv_schema   | materialized_view_5ef0d754 | Recompute
```

```
106     | 12727          | tickit_db     | ALTER TABLE ALTER SORTKEY   | 2023-07-26
15:00:08.006235 | tickit_db                 | mv_schema         | test_table_58102435 |
mv_schema   | materialized_view_ca746631 | Recompute
106     | 12720          | tickit_db     | Column was renamed          | 2023-07-26
14:59:12.82297  | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_a1f3f862 | Unrefreshable
106     | 12727          | tickit_db     | Table was renamed           | 2023-07-26
15:00:08.051321 | tickit_db                 | mv_schema         | test_table_95d6d861 |
mv_schema   | materialized_view_a1f3f862 | Unrefreshable
```

SYS_PROCEDURE_CALL

Utilisez la vue SYS_PROCEDURE_CALL pour obtenir des informations sur les appels de procédure stockée, notamment l'heure de début, l'heure de fin, le statut de l'appel de procédure stockée et la hiérarchie des appels pour les appels de procédure stockée imbriqués. Chaque appel de procédure stockée reçoit un ID de requête.

SYS_PROCEDURE_CALL est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
session_user_id	entier	Identifiant de l'utilisateur qui a créé la séance et qui est le demandeur de l'appel de procédure stockée de premier niveau.
security_user_id	entier	Identifiant de l'utilisateur dont les privilèges ont été utilisés pour exécuter l'instruction dans la procédure stockée. Si la procédure stockée a été appelée depuis la procédure stockée DEFINER externe, il s'agira du propriétaire user_id

Nom de la colonne	Type de données	Description
		de cette procédure stockée externe.
query_id	entier	Identifiant de requête de l'appel de procédure stockée.
query_text	char(4000)	Texte de la requête d'appel de procédure stockée.
start_time	timestamp	Heure UTC à laquelle la requête a commencé à s'exécuter. L'horodatage utilise six chiffres de précision pour des fractions de seconde, par exemple. 2009-06-12 11:29:19.131358.
end_time	timestamp	Heure UTC à laquelle la requête est terminée. L'horodatage utilise six chiffres de précision pour des fractions de seconde, par exemple : 2009-06-12 11:29:19.131358.
status	char(10)	Statut de l'appel de procédure stockée. Lorsque la procédure stockée est arrêtée par le système ou abandonnée par l'utilisateur, la valeur est abandonnée. Si l'appel de procédure stockée s'exécute jusqu'à son terme, la valeur est success.

Nom de la colonne	Type de données	Description
caller_procedure_query_id	entier	Si l'appel de procédure stockée a été appelé par un autre appel de procédure stockée, alors cette colonne contient l'ID de requête de l'appel extérieur. Sinon, le champ a pour valeur NULL.

Exemples de requêtes

La requête suivante renvoie une hiérarchie d'appels de procédure stockée imbriquée.

```
select query_id, datediff(seconds, start_time, end_time) as elapsed_time, status,
 trim(query_text) as call, caller_procedure_query_id from sys_procedure_call;
```

Exemple de sortie.

```
 query_id | elapsed_time | status  |                      call                      |
 caller_procedure_query_id
----------+--------------+---------+------------------------------------------------
+---------------------------
     3087 |           18 | success | CALL proc_bd906c98c45443ffa165e9552056902d(1)    |
          3085
     3085 |           18 | success | CALL proc_bd906c98c45443ffa165e9552056902d_2(1); |

(2 rows)
```

SYS_PROCEDURE_MESSAGES

SYS_PROCEDURE_MESSAGES est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
transaction_id	bigint	Identificateur de transaction.
query_id	entier	Identifiant de requête de l'appel de procédure stockée.
record_time	timestamp	Heure UTC à laquelle le message a été généré.
log_level	char(10)	Niveau de journalisation du message généré. Les valeurs possibles sont LOG, INFO, NOTICE, WARNING et EXCEPTION.
message	char(1024)	Texte du message généré.
line_number	entier	Numéro de ligne du message généré.

Exemples de requêtes

La requête suivante montre un exemple de sortie de SYS_PROCEDURE_MESSAGES.

```
select transaction_id, query_id, record_time, log_level, trim(message), line_number
 from sys_procedure_messages;
```

```
transaction_id | query_id |        record_time         | log_level |           btrim
        | line_number
---------------+----------+----------------------------+-----------
+--------------------------+-------------
     25267     |   80562  | 2023-07-17 14:38:31.910136 |   NOTICE  |
 test_notice_msg_b9f1e749  |     8
     25267     |   80562  | 2023-07-17 14:38:31.910002 |    LOG    |
 test_log_msg_833c7420     |     6
```

```
     25267     |   80562  | 2023-07-17 14:38:31.910111 |    INFO   |
 test_info_msg_651373d9   |     7
     25267     |   80562  | 2023-07-17 14:38:31.910154 |   WARNING |
 test_warning_msg_831c5747 |     9
(4 rows)
```

SYS_QUERY_DETAIL

Utilisez SYS_QUERY_DETAIL pour afficher les détails des requêtes à différents niveaux métriques, chaque ligne représentant les détails d'une requête WLM particulière à un niveau métrique donné. Cette vue contient de nombreux types de requêtes, comme les commandes DDL, DML et utilitaire (par exemple, copier et décharger). Certaines colonnes peuvent ne pas être pertinentes en fonction du type de requête. Par exemple, external_scanned_bytes n'est pas pertinent pour les tables internes.

SYS_QUERY_DETAIL est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> Pour vérifier si une transaction contenant la requête exécutée a été validée avec succès, vous devez effectuer une opération de jointure entre les tables système et la `sys_transaction_history` table. Par exemple :
>
> ```
> SELECT
>     th.transaction_id,
>     qd.query_id,
>     th.status AS transaction_status
> FROM
>     sys_query_detail qd
> LEFT JOIN sys_query_history qh ON qd.query_id = qh.query_id
> LEFT JOIN sys_transaction_history th on qh.transaction_id = th.transaction_id;
> ```

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identificateur de l'utilisateur qui a envoyé la requête.
query_id	bigint	Identificateur de requête.
child_query_séquence	entier	La séquence de la requête utilisateur réécrite, en commençant par 1.
stream_id	entier	Identificateur de flux du flux d'exécution de la requête.
segment_id	entier	Identificateur du segment d'exécution de la requête.
step_id	entier	Identificateur d'étape dans un segment.
step_name	text	Nom de l'étape dans un segment. Les valeurs possibles sont `aggregate broadcast ,delete,distribut e ,hash,hashjoin,insert,limit,me ,unique`, et`window`.
table_id	entier	Identificateur de table pour les analyses permanentes de table.
table_name	character(136)	Nom de la table de l'étape en cours d'opération.
is_rrscan	character	Valeur qui indique si une étape est une étape d'analyse

Nom de la colonne	Type de données	Description
		. La valeur true (t), indique qu'une analyse à plage restreinte a été utilisée.
start_time	timestamp	Heure à laquelle l'étape de requête a commencé. Ce champ est enregistré au niveau du segment, quelle que soit la valeur de la `metrics_level` colonne.
end_time	timestamp	Heure à laquelle l'étape de requête s'est terminée. Ce champ est enregistré au niveau du segment, quelle que soit la valeur de la `metrics_level` colonne.
duration	bigint	Temps (microsecondes) passé sur l'étape. Ce champ est enregistré au niveau du segment, quelle que soit la valeur de la `metrics_level` colonne.
alerte	text	Description de l'événement d'alerte.
input_bytes	bigint	Octets en entrée de l'étape en cours.
input_rows	bigint	Lignes d'entrée de l'étape en cours.
output_bytes	bigint	Octets de sortie de l'étape en cours.

Nom de la colonne	Type de données	Description
output_rows	bigint	Lignes de sortie de l'étape en cours.
blocks_read	bigint	Nombre de blocs lus par l'étape.
blocks_write	bigint	Nombre de blocs écrits par l'étape.
local_read_IO	bigint	Nombre de blocs lus depuis le cache du disque local.
remote_read_IO	bigint	Nombre de blocs lus à distance.
source	text	Type d'objet de base de données qui a été scanné. Cette colonne n'a de valeur que lorsque la valeur step_name de la ligne est `scan`.
data_skewness	entier	Asymétrie de la répartition des lignes de sortie entre toutes les étapes. Il s'agit d'un nombre compris entre 0 et 100 %. Plus le nombre est élevé, plus la répartition est déséquilibrée.

Nom de la colonne	Type de données	Description
time_skewness	entier	Asymétrie de la répartition du temps d'exécution entre toutes les étapes. Il s'agit d'un nombre compris entre 0 et 100 %. Plus le nombre est élevé, plus la répartition est déséquilibrée.
is_active	character	État de la requête au niveau de l'étape. Les valeurs possibles sont « t » (étape en cours d'exécution) ou « f » (étape en fin d'exécuti on).
spilled_block_local_disk	bigint	Nombre de blocs déversés sur le disque local.
spilled_block_remote_disk	bigint	Nombre de blocs déversés vers Amazon Simple Storage Service.
step_attribute	character(64)	Contient des informations concernant l'étape associée. Valeurs possibles pour les étapes d'analyse : `multi-dimensional` .

Nom de la colonne	Type de données	Description
metrics_level	character(64)	Le niveau métrique de la requête. Les valeurs possibles sont les suivantes : • requête pour enfant • stream • segment • step •
plan_parent_id	entier	Identifiant du nœud parent du nœud du plan. Un nœud parent peut avoir plusieurs nœuds enfants. Par exemple, une jointure par fusion est le nœud parent des scans sur les tables jointes.
plan_node_id	entier	Identifiant d'un nœud de plan mappé à une ou plusieurs étapes de la requête.

Notes d’utilisation

SYS_QUERY_DETAIL peut contenir des métriques au niveau de l'étape, du stream, du segment et de la requête enfant. En plus de référencer la colonne metrics_level, vous pouvez voir le niveau de métrique affiché par une ligne donnée en référençant les champs step_id, segment_id et stream_id conformément au tableau suivant.

Niveau métrique	valeur stream_id	valeur segment_id	valeur step_id
requête pour enfant	-1	-1	-1

Niveau métrique	valeur stream_id	valeur segment_id	valeur step_id
stream	Une valeur d'étape valide	-1	-1
segment	Une valeur d'étape valide	Une valeur d'étape valide	-1
step	Une valeur d'étape valide	Une valeur d'étape valide	Une valeur d'étape valide

Exemples de requêtes

L'exemple suivant renvoie la sortie de SYS_QUERY_DETAIL.

La requête suivante montre le détail des métadonnées de la requête au niveau de l'étape, y compris le nom de l'étape, input_bytes, output_bytes, input_rows, output_rows.

```
SELECT query_id,
       child_query_sequence,
       stream_id,
       segment_id,
       step_id,
       trim(step_name) AS step_name,
       duration,
       input_bytes,
       output_bytes,
       input_rows,
       output_rows
FROM sys_query_detail
WHERE query_id IN (193929)
ORDER BY query_id,
         stream_id,
         segment_id,
         step_id DESC;
```

Exemple de sortie.

```
 query_id | child_query_sequence | stream_id | segment_id | step_id | step_name  |
 duration     | input_bytes | output_bytes | input_rows | output_rows
```

```
----------+----------------------+-----------+------------+---------+------------
+-----------------+-------------+--------------+------------+-------------
   193929 |                    2 |         0 |          0 |       3 | hash       |
      37144 |           0 |      9350272 |          0 |      292196
   193929 |                    5 |         0 |          0 |       3 | hash       |
       9492 |           0 |        23360 |          0 |        1460
   193929 |                    1 |         0 |          0 |       3 | hash       |
      46809 |           0 |      9350272 |          0 |      292196
   193929 |                    4 |         0 |          0 |       2 | return     |
       7685 |           0 |          896 |          0 |         112
   193929 |                    1 |         0 |          0 |       2 | project    |
      46809 |           0 |            0 |          0 |      292196
   193929 |                    2 |         0 |          0 |       2 | project    |
      37144 |           0 |            0 |          0 |      292196
   193929 |                    5 |         0 |          0 |       2 | project    |
       9492 |           0 |            0 |          0 |        1460
   193929 |                    3 |         0 |          0 |       2 | return     |
      11033 |           0 |        14336 |          0 |         112
   193929 |                    2 |         0 |          0 |       1 | project    |
      37144 |           0 |            0 |          0 |      292196
   193929 |                    1 |         0 |          0 |       1 | project    |
      46809 |           0 |            0 |          0 |      292196
   193929 |                    5 |         0 |          0 |       1 | project    |
       9492 |           0 |            0 |          0 |        1460
   193929 |                    3 |         0 |          0 |       1 | aggregate  |
      11033 |           0 |       201488 |          0 |          14
   193929 |                    4 |         0 |          0 |       1 | aggregate  |
       7685 |           0 |        28784 |          0 |          14
   193929 |                    5 |         0 |          0 |       0 | scan       |
       9492 |           0 |        23360 |     292196 |        1460
   193929 |                    4 |         0 |          0 |       0 | scan       |
       7685 |           0 |         1344 |        112 |         112
   193929 |                    2 |         0 |          0 |       0 | scan       |
      37144 |           0 |      7304900 |     292196 |      292196
   193929 |                    3 |         0 |          0 |       0 | scan       |
      11033 |           0 |        13440 |        112 |         112
   193929 |                    1 |         0 |          0 |       0 | scan       |
      46809 |           0 |      7304900 |     292196 |      292196
   193929 |                    5 |         0 |          0 |      -1 |            |
       9492 |       12288 |            0 |          0 |          0
   193929 |                    1 |         0 |          0 |      -1 |            |
      46809 |       16384 |            0 |          0 |          0
   193929 |                    2 |         0 |          0 |      -1 |            |
      37144 |       16384 |            0 |          0 |          0
```

```
   193929 |                   4 |         0 |          0 |      -1 |           |
     7685 |       28672 |           0 |          0 |          0
   193929 |                   3 |         0 |          0 |      -1 |           |
    11033 |      114688 |           0 |          0 |          0
```

Pour afficher les tables de votre base de données dans l'ordre, de la plus utilisée à la moins utilisée, utilisez l'exemple suivant. *sample_data_dev*Remplacez-le par votre propre base de données. Notez que cette requête compte les requêtes à partir de la création de votre cluster, mais que les données de votre vue système ne sont pas enregistrées lorsque votre entrepôt des données manque d'espace.

```
SELECT table_name, COUNT (DISTINCT query_id)
FROM SYS_QUERY_DETAIL
WHERE table_name LIKE 'sample_data_dev%'
GROUP BY table_name
ORDER BY COUNT(*) DESC;

+---------------------------------+-------+
|           table_name            | count |
+---------------------------------+-------+
| sample_data_dev.tickit.venue    |     4 |
| sample_data_dev.myunload1.venue |     3 |
| sample_data_dev.tickit.listing  |     1 |
| sample_data_dev.tickit.category |     1 |
| sample_data_dev.tickit.users    |     1 |
| sample_data_dev.tickit.date     |     1 |
| sample_data_dev.tickit.sales    |     1 |
| sample_data_dev.tickit.event    |     1 |
+---------------------------------+-------+
```

L'exemple suivant montre les différents niveaux de mesure pour une seule requête WLM.

```
SELECT query_id, child_query_sequence, stream_id, segment_id, step_id, step_name,
 start_time, end_time, metrics_level
FROM sys_query_detail
WHERE query_id = 1553 AND step_id = -1
ORDER BY stream_id, segment_id, step_id;

 query_id | child_query_sequence | stream_id | segment_id | step_id | step_name |
   start_time         |          end_time          | metrics_level
----------+----------------------+-----------+------------+---------+-----------
+----------------------------+----------------------------+---------------
```

```
    1553 |                    1 |        -1 |         -1 |      -1 |           |
 2024-10-17 02:28:49.814721 | 2024-10-17 02:28:49.847838 | child query
    1553 |                    1 |         0 |         -1 |      -1 |           |
 2024-10-17 02:28:49.814721 | 2024-10-17 02:28:49.835609 | stream
    1553 |                    1 |         0 |          0 |      -1 |           |
 2024-10-17 02:28:49.824677 | 2024-10-17 02:28:49.830372 | segment
    1553 |                    1 |         1 |         -1 |      -1 |           |
 2024-10-17 02:28:49.835624 | 2024-10-17 02:28:49.845773 | stream
    1553 |                    1 |         1 |          1 |      -1 |           |
 2024-10-17 02:28:49.84088  | 2024-10-17 02:28:49.842388 | segment
    1553 |                    1 |         1 |          2 |      -1 |           |
 2024-10-17 02:28:49.835926 | 2024-10-17 02:28:49.844396 | segment
    1553 |                    1 |         2 |         -1 |      -1 |           |
 2024-10-17 02:28:49.846949 | 2024-10-17 02:28:49.847838 | stream
    1553 |                    1 |         2 |          3 |      -1 |           |
 2024-10-17 02:28:49.847013 | 2024-10-17 02:28:49.847485 | segment
(8 rows)
```

SYS_QUERY_EXPLAIN

Affiche le plan EXPLAIN pour une requête qui a été soumise à exécution.

SYS_QUERY_EXPLAIN est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identificateur de l'utilisateur qui a envoyé la requête.
query_id	bigint	Identificateur de requête. Les informations de requête détaillées sont stockées dansSYS_QUERY_HISTORY.
child_query_séquence	entier	La séquence de la requête utilisateur réécrite, en commençant par 1.

Nom de la colonne	Type de données	Description
plan_node_id	entier	Identifiant d'un nœud de plan mappé à une ou plusieurs étapes de la requête.
plan_parent_id	entier	Identifiant du nœud parent du nœud du plan. Un nœud parent peut avoir plusieurs nœuds enfants. Par exemple, une jointure par fusion est le nœud parent des scans sur les tables jointes.
plan_node	character(400)	Texte du nœud de la sortie EXPLAIN. Les nœuds de plan qui font référence à l'exécution sur des nœuds de calcul sont préfixés par XN dans la sortie EXPLAIN.
info_nœud	character(400)	Informations de qualificateur et de filtre pour le nœud de plan. Par exemple, les conditions de jointure et les restrictions clause WHERE sont incluses dans cette colonne.

Exemples de requêtes

L'exemple suivant est le plan EXPLAIN d'une seule requête.

```
SELECT * FROM sys_query_explain WHERE query_id = 612635 ORDER_BY plan_node_id;

 userid | query_id | child_query_sequence | plan_node_id | plan_parent_id |


          plan_node
```

```
                                          |

                                                                plan_info



--------+----------+----------------------+--------------+----------------
+-----------------------------------------------------------------------------------------------
+-----------------------------------------------------------------------------------------------
    100 |   612635 |                    1 |            1 |              0 | XN Limit
 (cost=3604047533041.00..3604047533041.25 rows=100 width=20)



                                            |





    100 |   612635 |                    1 |            2 |              1 |   ->  XN
 Merge  (cost=3604047533041.00..3604047533148.02 rows=42809 width=20)



                                             | Merge Key: sum(b.totalprice)





    100 |   612635 |                    1 |            3 |              2 |         ->
 XN Network  (cost=3604047533041.00..3604047533148.02 rows=42809 width=20)



                                          |   Send to leader
```

```
   100 |   612635 |                    1 |           4 |             3 |
  ->  XN Sort  (cost=3604047533041.00..3604047533148.02 rows=42809 width=20)

                                        | Sort Key: sum(b.totalprice)

   100 |   612635 |                    1 |           5 |             4 |
          ->  XN HashAggregate  (cost=2604047529640.76..2604047529747.78 rows=42809
width=20)

                                              |

   100 |   612635 |                    1 |           6 |             5 |
                ->  XN Hash Join DS_DIST_NONE  (cost=15104956.16..2602364653507.34
rows=336575226684 width=20)

                                                    | Hash Cond: (("outer".listid =
"inner".listid) AND ("outer".sellerid = "inner".sellerid))

   100 |   612635 |                    1 |           7 |             6 |
                           ->  XN Seq Scan on listing b  (cost=0.00..7884677.12
rows=788467712 width=24)

                                                     |
```

```
    100 |   612635 |                    1 |            8 |              6 |
                   ->  XN Hash  (cost=7063797.76..7063797.76 rows=706379776 width=8)

                                         |

    100 |   612635 |                    1 |            9 |              8 |
                              ->  XN Seq Scan on sales a  (cost=0.00..7063797.76
 rows=706379776 width=8)

                                                  |

(9 rows)
```

SYS_QUERY_HISTORY

Utilisez SYS_QUERY_HISTORY pour afficher les détails des requêtes des utilisateurs. Chaque ligne représente une requête utilisateur avec des statistiques cumulées pour certains des champs. Cette vue contient de nombreux types de requêtes, tels que DDL (Data Definition Language), DML (Data Manipulation Language), copie, déchargement et Amazon Redshift Spectrum. Elle contient à la fois les requêtes en cours d'exécution et terminées.

SVV_QUERY_STATE est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> Pour vérifier si une transaction contenant la requête exécutée a été validée avec succès, vous devez effectuer une opération de jointure entre les tables système et la `sys_transaction_history` table. Par exemple :

```
SELECT
    qh.transaction_id,
    qh.query_id,
    qh.status AS query_status,
    qh.query_type,
    TRIM(qh.query_text) AS query_text,
    th.status AS transaction_status
FROM
    sys_query_history qh
LEFT JOIN
    sys_transaction_history th ON qh.transaction_id = th.transaction_id;
```

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identificateur de l'utilisateur qui a envoyé la requête.
query_id	bigint	Identificateur de requête.
query_label	caractère (320)	Nom abrégé de la requête.
transaction_id	bigint	Identificateur de transaction.
session_id	entier	Identificateur de processus du processus exécutant la requête.
database_name	character(128)	Nom de la base de données à laquelle l'utilisateur était

Nom de la colonne	Type de données	Description
		connecté lorsque la requête a été émise.
query_type	character(32)	Type de requête, tel que SELECT, INSERT, UPDATE, UNLOAD, COPY, COMMAND, DDL, UTILITY, CTAS et OTHER.
status	character(10)	Statut de la requête. Valeurs valides : planification, mise en file d'attente, exécution, retour, échec, annulation et succès.
result_cache_hit	Booléen	Indique si la requête correspond au cache de résultats.
start_time	timestamp	Heure à laquelle la requête a commencé.
end_time	timestamp	Heure à laquelle la requête s'est terminée.
elapsed_time	bigint	Temps total (microsecondes) passé sur la requête.
queue_time	bigint	Temps total (microsecondes) passé dans la file d'attente de requêtes de classe de service.
execution_time	bigint	Durée totale (microsecondes) d'exécution dans la classe de service.
error_message	character(512)	Raison pour laquelle une requête a échoué.

Nom de la colonne	Type de données	Description
returned_rows	bigint	Nombre de lignes retournées au client.
returned_bytes	bigint	Nombre d'octets retournés au client.
query_text	character(4000)	Chaîne de requête. Cette chaîne peut être tronquée.
redshift_version	character(256)	Version Amazon Redshift lors de l'exécution de la requête.
usage_limit	character(150)	Liste des limites d'utilisation IDs atteintes par la requête.
compute_type	varchar(32)	Indique si la requête s'exécute sur le cluster principal ou sur le cluster de mise à l'échelle de simultanéité. Les valeurs possibles sont `primary` (la requête s'exécute sur le cluster principal), `secondary` (la requête s'exécute sur le cluster secondaire), ou `primary-scale` (la requête s'exécute sur le cluster de simultanéité). Cela ne s'applique qu'au cluster provisionné.
compile_time	bigint	Le temps total (en microsecondes) consacré à la compilation de la requête.

Nom de la colonne	Type de données	Description
planning_time	bigint	Le temps total (en microsecondes) consacré à la planification de la requête.
lock_wait time	bigint	Le temps total (microsecondes) passé à attendre le verrou relationnel.
identifiant_classe de service	entier	L'ID de la classe de service. Pour obtenir la liste des classes de service IDs, rendez-vous surClasse de service WLM IDs. Cette colonne est uniquement utilisée pour les requêtes exécutées sur des clusters provisionnés. Pour les requêtes exécutées sur Redshift Serverless, cette colonne contient -1.
service_class_name	character(64)	Nom de la classe de service. Cette colonne est uniquement utilisée pour les requêtes exécutées sur des clusters provisionnés. Pour les requêtes exécutées sur Amazon Redshift Redshift Serverless, cette colonne est vide.

Nom de la colonne	Type de données	Description
query_priority	character(20)	Priorité de la file d'attente dans laquelle la requête a été exécutée. Les valeurs possibles sont les suivantes : • NULL • le plus bas • bas • normal • haut • le plus élevé NULL signifie que la priorité de requête n'est pas prise en charge pour la requête. Cette colonne est uniquement utilisée pour les requêtes exécutées sur des clusters provisionnés. Pour les requêtes exécutées sur Redshift Serverless, cette colonne est vide.

Nom de la colonne	Type de données	Description
requête_courte accélérée	character(10)	Si la requête a été accélérée à l'aide de l'accélération des requêtes courtes (SQA). Les valeurs possibles sont les suivantes : • vrai • false • NULL Cette colonne est uniquement utilisée pour les requêtes exécutées sur des clusters provisionnés. Pour les requêtes exécutées sur Redshift Serverless, cette colonne est vide.
user_query_hash	character(40)	Le hachage de la requête généré à partir de la requête, y compris ses littéraux de requête. Les requêtes répétées avec le même texte de requête auront les mêmes valeurs user_query_hash.

Nom de la colonne	Type de données	Description
generic_query_hash	character(40)	Le hachage de la requête généré à partir de la requête, à l'exclusion de ses littéraux de requête. Les requêtes répétées avec le même texte de requête, mais des littéraux de requête différents, auront les mêmes valeurs generic_query_hash.
version de hachage de la requête	entier	Le numéro de version du hachage de requête généré à partir de la requête.
Result_cache_query_id	entier	Si la requête a utilisé la mise en cache des résultats, cette valeur de champ est l'ID de requête à l'origine des résultats mis en cache. Si la mise en cache des résultats n'a pas été utilisée, cette valeur de champ est 0.
nom d'utilisateur	character(128)	Le nom d'utilisateur de l'utilisateur qui a soumis la requête.

Exemples de requêtes

La requête suivante renvoie les requêtes en cours d'exécution et en file d'attente.

```
SELECT user_id,
       query_id,
       transaction_id,
       session_id,
       status,
       trim(database_name) AS database_name,
```

```
       start_time,
       end_time,
       result_cache_hit,
       elapsed_time,
       queue_time,
       execution_time
FROM sys_query_history
WHERE status IN ('running','queued')
ORDER BY start_time;
```

Exemple de sortie.

```
 user_id | query_id | transaction_id | session_id | status  | database_name |
 start_time         |          end_time          | result_cache_hit | elapsed_time |
 queue_time | execution_time
---------+----------+----------------+------------+---------+---------------
+--------------------------+--------------------------+------------------
+-------------+------------+----------------
     101 |   760705 |         852337 | 1073832321 | running | tpcds_1t      |
 2022-02-15 19:03:19.67849 | 2022-02-15 19:03:19.739811 | f               |
 61321 |          0 |              0
```

La requête suivante renvoie l'heure de début, l'heure de fin, la durée de la file d'attente, le temps écoulé, le temps de planification et et d'autres métadonnées pour une requête spécifique.

```
SELECT user_id,
       query_id,
       transaction_id,
       session_id,
       status,
       trim(database_name) AS database_name,
       start_time,
       end_time,
       result_cache_hit,
       elapsed_time,
       queue_time,
       execution_time,
       planning_time,
       trim(query_text) as query_text
FROM sys_query_history
WHERE query_id = 3093;
```

Exemple de sortie.

```
user_id | query_id | transaction_id | session_id |   status   | database_name |
   start_time         |          end_time          | result_cache_hit | elapsed_time |
 queue_time | execution_time | planning_time | query_text
--------+----------+----------------+------------+------------
+---------------+----------------------------+----------------------------
+-----------------+-------------+-----------+---------------+--------------
+-------------------------------------
    106 |     3093 |          11759 | 1073750146 | success    | dev           |
 2023-03-16 16:53:17.840214 | 2023-03-16 16:53:18.106588 | f                |
 266374 |          0 |         105725 |        136589 | select count(*) from item;
```

La requête suivante répertorie les 10 requêtes SELECT les plus récentes.

```
SELECT query_id,
       transaction_id,
       session_id,
       start_time,
       elapsed_time,
       queue_time,
       execution_time,
       returned_rows,
       returned_bytes
FROM sys_query_history
WHERE query_type = 'SELECT'
ORDER BY start_time DESC limit 10;
```

Exemple de sortie.

```
 query_id | transaction_id | session_id |         start_time         | elapsed_time |
 queue_time | execution_time | returned_rows | returned_bytes
----------+----------------+------------+----------------------------+--------------
+------------+----------------+---------------+----------------
   526532 |          61093 | 1073840313 | 2022-02-09 04:43:24.149603 |       520571 |
        0 |         481293 |             1 |           3794
   526520 |          60850 | 1073840313 | 2022-02-09 04:38:27.24875  |       635957 |
        0 |         596601 |             1 |           3679
   526508 |          60803 | 1073840313 | 2022-02-09 04:37:51.118835 |       563882 |
        0 |         503135 |             5 |          17216
   526505 |          60763 | 1073840313 | 2022-02-09 04:36:48.636224 |       649337 |
        0 |         589823 |             1 |            652
```

```
   526478 |          60730 | 1073840313 | 2022-02-09 04:36:11.741471 |     14611321 |
        0 |       14544058 |             0 |              0
   526467 |          60636 | 1073840313 | 2022-02-09 04:34:11.91463  |     16711367 |
        0 |       16633767 |             1 |            575
   511617 |         617946 | 1074009948 | 2022-01-20 06:21:54.44481  |      9937090 |
        0 |        9899271 |           100 |          12500
   511603 |         617941 | 1074259415 | 2022-01-20 06:21:45.71744  |      8065081 |
        0 |        7582500 |           100 |           8889
   511595 |         617935 | 1074128320 | 2022-01-20 06:21:44.030876 |      1051270 |
        0 |        1014879 |             1 |             72
   511584 |         617931 | 1074030019 | 2022-01-20 06:21:42.764088 |       609033 |
        0 |         485887 |           100 |           8438
```

La requête suivante indique le nombre quotidien de requêtes de sélection et le temps moyen écoulé pour les requêtes.

```
SELECT date_trunc('day',start_time) AS exec_day,
       status,
       COUNT(*) AS query_cnt,
       AVG(datediff (microsecond,start_time,end_time)) AS elapsed_avg
FROM sys_query_history
WHERE query_type = 'SELECT'
AND start_time >= '2022-01-14'
AND start_time <= '2022-01-18'
GROUP BY exec_day,
         status
ORDER BY exec_day,
         status;
```

Exemple de sortie.

```
      exec_day       | status  | query_cnt | elapsed_avg
---------------------+---------+-----------+------------
 2022-01-14 00:00:00 | success |      5253 |  56608048
 2022-01-15 00:00:00 | success |      7004 |  56995017
 2022-01-16 00:00:00 | success |      5253 |  57016363
 2022-01-17 00:00:00 | success |      5309 |  55236784
 2022-01-18 00:00:00 | success |      8092 |  54355124
```

La requête suivante indique la performance quotidienne du temps moyen écoulé pour les requêtes.

```
SELECT distinct date_trunc('day',start_time) AS exec_day,
```

```
       query_count.cnt AS query_count,
       Percentile_cont(0.5) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P50_runtime,
       Percentile_cont(0.8) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P80_runtime,
       Percentile_cont(0.9) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P90_runtime,
       Percentile_cont(0.99) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS P99_runtime,
       Percentile_cont(1.0) within group(ORDER BY elapsed_time) OVER (PARTITION BY
 exec_day) AS max_runtime
FROM sys_query_history
LEFT JOIN (SELECT  date_trunc('day',start_time) AS day, count(*) cnt
           FROM sys_query_history
           WHERE query_type = 'SELECT'
           GROUP by 1) query_count
ON date_trunc('day',start_time) = query_count.day
WHERE query_type = 'SELECT'
ORDER BY exec_day;
```

Exemple de sortie.

```
      exec_day       | query_count | p50_runtime | p80_runtime | p90_runtime |
 p99_runtime  | max_runtime
--------------------+-------------+-------------+-------------+-------------
+-------------+-------------
 2022-01-14 00:00:00 |        5253 |  16816922.0 |  69525096.0 | 158524917.8 |
 486322477.52 | 1582078873.0
 2022-01-15 00:00:00 |        7004 |  15896130.5 |  71058707.0 | 164314568.9 |
 500331542.07 | 1696344792.0
 2022-01-16 00:00:00 |        5253 |  15750451.0 |  72037082.2 | 159513733.4 |
 480372059.24 | 1594793766.0
 2022-01-17 00:00:00 |        5309 |  15394513.0 |  68881393.2 | 160254700.0 |
 493372245.84 | 1521758640.0
 2022-01-18 00:00:00 |        8092 |  15575286.5 |  68485955.4 | 154559572.5 |
 463552685.39 | 1542783444.0
 2022-01-19 00:00:00 |        5860 |  16648747.0 |  72470482.6 | 166485138.2 |
 492038228.67 | 1693483241.0
 2022-01-20 00:00:00 |        1751 |  15422072.0 |  69686381.0 | 162315385.0 |
 497066615.00 | 1439319739.0
 2022-02-09 00:00:00 |          13 |   6382812.0 |  17616161.6 |  21197988.4 |
 23021343.84 |   23168439.0
```

La requête suivante indique la distribution du type de requête.

```
SELECT query_type,
       COUNT(*) AS query_count
FROM sys_query_history
GROUP BY query_type
ORDER BY query_count DESC;
```

Exemple de sortie.

```
 query_type | query_count
------------+-------------
 UTILITY    |      134486
 SELECT     |       38537
 DDL        |        4832
 OTHER      |         768
 LOAD       |         768
 CTAS       |         748
 COMMAND    |          92
```

L'exemple suivant montre la différence entre les résultats de hachage des requêtes entre plusieurs requêtes. Observez les requêtes suivantes :

```
CREATE TABLE test_table (col1 INT);

INSERT INTO test_table VALUES (1),(2);

SELECT * FROM test_table;

SELECT * FROM test_table;

SELECT col1 FROM test_table;

SELECT * FROM test_table WHERE col1=1;

SELECT * FROM test_table WHERE col1=2;

SELECT query_id, TRIM(user_query_hash) AS user_query_hash, TRIM(generic_query_hash)
 AS generic_query_hash, TRIM(query_text) AS text FROM sys_query_history ORDER BY
 start_time
DESC LIMIT 10;
```

Voici un exemple de sortie :

```
query_id | user_query_hash | generic_query_hash | text
---------+-----------------+--------------------+----------
24723049 | oPuFtjEPLTs=    | oPuFtjEPLTs=       | select query_id,
 trim(user_query_hash) as user_query_hash, trim(generic_query_hash) as
 generic_query_hash, query_hash_version, trim(query_text) as text from
 sys_query_history order by start_time\r\ndesc limit 20
24723045 | Gw2Kwdd8m2I=    | IwfRu8/XAKI=       | select * from test_table where col1=2
 limit 100
24723041 | LNw2vx0GDXo=    | IwfRu8/XAKI=       | select * from test_table where col1=1
 limit 100
24723036 | H+qep/c82Y8=    | H+qep/c82Y8=       | select col1 from test_table limit 100
24723033 | H+qep/c82Y8=    | H+qep/c82Y8=       | select * from test_table limit 100
24723029 | H+qep/c82Y8=    | H+qep/c82Y8=       | select * from test_table limit 100
24723023 | 50sirx9E1hU=    | uO36Z1a/QYs=       | insert into test_table values (1),(2)
24723021 | YSVnlivZHeo=    | YSVnlivZHeo=       | create table test_table (col1 int)
```

`SELECT * FROM test_table;` et `SELECT col1 FROM test_table;` ont la même valeur user_query_hash, puisque test_table n'a qu'une seule colonne. `SELECT * FROM test_table WHERE col1=1;` et `SELECT * FROM test_table WHERE col1=2;` ont des valeurs user_query_hash différentes, mais des valeurs generic_query_hash identiques, car les deux requêtes sont identiques en dehors des littéraux de requête 1 et 2.

SYS_QUERY_TEXT

Utilisez SYS_QUERY_TEXT pour afficher le texte de toutes les requêtes. Chaque ligne représente le texte des requêtes de 4 000 caractères maximum en commençant par le numéro de séquence 0. Lorsque l'instruction de requête contient plus de 4 000 caractères, des lignes supplémentaires sont enregistrées pour l'instruction en incrémentant le numéro de séquence pour chaque ligne. Cette vue journalise tous les textes des requêtes des utilisateurs, tels que les requêtes DDL, utilitaires, Amazon Redshift et les requêtes réservées au nœud leader.

SYS_QUERY_TEXT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identificateur de l'utilisateur qui a envoyé la requête.
query_id	bigint	Identificateur de requête.
transaction_id	bigint	Identificateur de la transaction associé à l'instruction.
session_id	entier	Identificateur du processus de la session exécutant la requête.
start_time	timestamp	Moment auquel la requête démarre.
sequence	entier	Lorsqu'une seule instruction contient plus de 4 000 caractères, des lignes supplémentaires sont enregistrées pour l'instruction. Sequence 0 correspond à la première ligne, 1 à la deuxième ligne, et ainsi de suite.
text	character(4000)	Le texte de la requête SQL par incréments de 4 000 caractères. Ce champ peut contenir des caractères spéciaux tels qu'une barre oblique inverse (\) et un caractère de saut de ligne (\n).

Exemples de requêtes

La requête suivante renvoie les requêtes en cours d'exécution et en file d'attente.

```
SELECT user_id,
 query_id,
 transaction_id,
 session_id, start_time,
 sequence, trim(text) as text from sys_query_text
 ORDER BY sequence;
```

Exemple de sortie.

```
 user_id | query_id | transaction_id | session_id |        start_time          |
 sequence |                                                        text
--------+----------+-----------------+------------+----------------------------
+----------
+------------------------------------------------------------------------------------------------
   100  |     4    |       1396      | 1073750220 | 2023-04-28 16:44:55.887184 |
   0    | SELECT trim(text) as text, sequence FROM sys_query_text WHERE query_id =
 pg_last_query_id() AND user_id > 1 AND start
_time > '2023-04-28 16:44:55.922705+00:00'::timestamp order by sequence;
```

La requête suivante renvoie les autorisations accordées ou révoquées des groupes de votre base de données.

```
SELECT
    SPLIT_PART(text, ' ', 1) as grantrevoke,
    SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), 'GROUP'))), ' ', 2) as group,
    SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), ' '))), 'ON', 1) as type,
    SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), 'ON'))), ' ', 2) || ' ' ||
 SPLIT_PART((SUBSTRING(text, STRPOS(UPPER(text), 'ON'))), ' ', 3) as entity
FROM SYS_QUERY_TEXT
WHERE (text LIKE 'GRANT%' OR text LIKE 'REVOKE%') AND text LIKE '%GROUP%';

+-------------+----------+--------+----------+
| grantrevoke |  group   |  type  |  entity  |
+-------------+----------+--------+----------+
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | USAGE  | TABLE t1 |
```

```
| GRANT       | bi_group | SELECT | TABLE t1 |
| GRANT       | bi_group | SELECT | TABLE t1 |
+-------------+----------+--------+----------+
```

SYS_RESTORE_LOG

Utilisez SYS_RESTORE_LOG pour surveiller la progression de la migration de chaque table du cluster lors d'un redimensionnement classique vers des nœuds. RA3 Elle capture le débit historique de la migration des données lors de l'opération de redimensionnement. Pour plus d'informations sur le redimensionnement classique en RA3 nœuds, consultez la section Redimensionnement classique.

SYS_RESTORE_STATE n'est visible que par les super-utilisateurs.

Colonnes de la table

Nom de la colonne	Type de données	Description
event_time	timestamp	Horodatage qui indique le moment où l'entrée de journal est enregistrée.
database_name	char(128)	Nom de la base de données.
nom_schéma	char(128)	Nom du schéma.
table_name	char(128)	Nom de la table.
table_id	entier	ID de la table.
action	char(128)	Action effectuée au moment de l'entrée. Les valeurs peuvent inclure : migration démarrée, point de contrôle, reprise, terminée, annulée ou réinitialisée.
table_size	long	Taille de la table.

Nom de la colonne	Type de données	Description
total_data_processed	long	Taille des données en Mo traitées jusqu'à présent pour la table.
delta_data_processed	long	Taille des données traitées depuis la dernière mise à jour d'event_time, en Mo. Cela vous permet de déterminer la quantité de données traitées depuis le dernier intervalle de temps enregistré. Vous pouvez la comparer à la valeur table_size pour avoir une idée de la rapidité du traitement des données.
message	char(512)	Explication détaillée de la valeur de la colonne Action.
redistribution_type	char(32)	Type de redistribution de la table. Soit une conversio n KEY, soit une tâche de rééquilibrage EVEN. Pour en savoir plus sur les styles de distribution, consultez Styles de Distribution.

Exemples de requêtes

La requête suivante calcule le débit du traitement des données à l'aide de SYS_RESTORE_LOG.

```
SELECT
    ROUND(sum(delta_data_processed) / 1024.0, 2) as data_processed_gb,
    ROUND(datediff(sec, min(event_time), max(event_time)) / 3600.0, 2) as duration_hr,
    ROUND(data_processed_gb/duration_hr, 2) as throughput_gb_per_hr
```

```
from sys_restore_log;
```

Exemple de sortie.

```
 data_processed_gb | duration_hr | throughput_gb_per_hr
-------------------+-------------+----------------------
              0.91 |        8.37 |                 0.11
(1 row)
```

La requête suivante qui montre tous les types de redistribution.

```
SELECT * from sys_restore_log ORDER BY event_time;
```

```
 database_name |     schema_name      |      table_name      | table_id |
 action             | total_data_processed | delta_data_processed |         event_time
        | table_size | message |   redistribution_type
---------------+----------------------+----------------------+----------
+-----------------------------+----------------------+----------------------
+-----------------------------+------------+---------+--------------------------
 dev           | schemaaaa877096d844d | customer_key         |   106424 |
 Redistribution started      |                    0 |                      | 2024-01-05
 02:18:00.744977 |        325 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | dp30907_t2_autokey   |   106430 |
 Redistribution started      |                    0 |                      | 2024-01-05
 02:18:02.756675 |         90 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | dp30907_t2_autokey   |   106430 |
 Redistribution completed    |                   90 |                   90 | 2024-01-05
 02:23:30.643718 |         90 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | customer_key         |   106424 |
 Redistribution completed    |                  325 |                  325 | 2024-01-05
 02:23:45.998249 |        325 |         | Restore Distkey Table
 dev           | schemaaaa877096d844d | dp30907_t1_even      |   106428 |
 Redistribution started      |                    0 |                      | 2024-01-05
 02:23:46.083849 |         30 |         | Rebalance Disteven Table
 dev           | schemaaaa877096d844d | dp30907_t5_auto_even |   106436 |
 Redistribution started      |                    0 |                      | 2024-01-05
 02:23:46.855728 |         45 |         | Rebalance Disteven Table
 dev           | schemaaaa877096d844d | dp30907_t5_auto_even |   106436 |
 Redistribution completed    |                   45 |                   45 | 2024-01-05
 02:24:16.343029 |         45 |         | Rebalance Disteven Table
```

```
 dev           | schemaaaa877096d844d | dp30907_t1_even      |   106428 |
 Redistribution completed    |                  30 |                  30 | 2024-01-05
 02:24:20.584703 |         30 |         | Rebalance Disteven Table
 dev           | schemaefd028a2a48a4c | customer_even        |   130512 |
 Redistribution started      |                   0 |                     | 2024-01-05
 04:54:55.641741 |        190 |         | Restore Disteven Table
 dev           | schemaefd028a2a48a4c | customer_even        |   130512 |
 Redistribution checkpointed |     29.4342113157737 |     29.4342113157737 | 2024-01-05
 04:55:04.770696 |        190 |         | Restore Disteven Table
(8 rows)
```

SYS_RESTORE_STATE

Utilisez SYS_RESTORE_STATE pour surveiller la progression de la migration de chaque table lors d'un redimensionnement classique. Cela s'applique spécifiquement lorsque le type de nœud cible est RA3. Pour plus d'informations sur le redimensionnement classique en RA3 nœuds, consultez la section Redimensionnement classique.

Seuls les super-utilisateurs peuvent voir SYS_RESTORE_STATE. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identificateur de l'utilisateur qui a envoyé la requête.
database_name	char(64)	Nom de la base de données de la table.
schema_id	entier	ID de schéma de la table.
table_id	entier	ID de la table.
table_name	char(128)	Nom de la table.
redistribution_status	char(128)	Statut de progression de la redistribution de la table. Les valeurs possibles sont

Nom de la colonne	Type de données	Description
		`Completed` , `In progress` et `Pending`.
percentage_redistributed	float	Pourcentage de progression de la redistribution de la table. Les valeurs possibles vont de 0 à 100 %. Par exemple, une valeur de 25 indique que 25 % des données sont redistrib uées.
redistribution_type	char(32)	Type de redistribution de la table. Soit une conversio n KEY, soit une tâche de rééquilibrage EVEN. Pour en savoir plus sur les styles de distribution, consultez Styles de Distribution.

Exemples de requêtes

La requête suivante renvoie des enregistrements pour les requêtes en cours d'exécution et les requêtes mises en file d'attente.

```
SELECT * FROM sys_restore_state;
```

Exemple de sortie.

```
 userid | database_name | schema_id | table_id |   table_name   | redistribution_status
 | precentage_redistributed |   redistribution_type
--------+---------------+-----------+----------+----------------
+----------------------+--------------------------+------------------------
    1   |     test1     |   124865  |  124878  | customer_key_4 |         Pending
 |      0                 |  Rebalance Disteven Table
    1   |      dev      |   124865  |  124874  | customer_key_3 |         Pending
 |      0                 |  Rebalance Disteven Table
```

```
    1   |      dev      |   124865  |  124870  | customer_key_2 |        Completed
 |     100                |  Rebalance Disteven Table
    1   |      dev      |   124865  |  124866  | customer_key_1 |       In progress
 |     13.52              |  Restore Distkey Table
```

Vous trouverez ci-dessous l'état du traitement des données.

```
SELECT
    redistribution_status, ROUND(SUM(block_count) / 1024.0, 2) AS total_size_gb
FROM sys_restore_state sys inner join stv_tbl_perm stv
    on sys.table_id = stv.id
GROUP BY sys.redistribution_status;
```

Exemple de sortie.

```
 redistribution_status | total_size_gb
-----------------------+---------------
 Completed             |          0.07
 Pending               |          0.71
 In progress           |          0.20
(3 rows)
```

SYS_SCHEMA_QUOTA_VIOLATIONS

Enregistre l’occurrence, l’ID de transaction et d’autres informations utiles lorsqu’un quota de schéma est dépassé. Cette table système est une traduction de STL_SCHEMA_QUOTA_VIOLATIONS.

r_SYS_SCHEMA_QUOTA_VIOLATIONS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
owner_id	entier	ID du propriétaire du schéma.

Nom de la colonne	Type de données	Description
user_id	entier	ID de l'utilisateur qui a généré l'entrée.
transaction_id	bigint	ID de transaction associé à l'instruction.
session_id	entier	ID de processus associé à l'instruction.
schema_id	entier	ID d'espace de noms ou de schéma.
nom_schéma	caractère (128)	Nom de l'espace de noms ou du schéma.
quota	entier	Quantité d'espace disque (en Mo) que le schéma peut utiliser.
disk_usage	entier	Espace disque (en Mo) actuellement utilisé par le schéma.
record_time	horodatage sans fuseau horaire	Heure à laquelle la violation s'est produite.

Exemples de requêtes

La requête suivante affiche le résultat d'une violation de quota :

```
SELECT user_id, TRIM(schema_name) "schema_name", quota, disk_usage, record_time FROM
sys_schema_quota_violations WHERE SCHEMA_NAME = 'sales_schema' ORDER BY timestamp DESC;
```

Cette requête renvoie l'exemple de sortie suivant pour le schéma spécifié :

```
user_id| schema_name  | quota | disk_usage | record_time
-------+--------------+-------+------------+----------------------------
104    | sales_schema | 2048  | 2798       | 2020-04-20 20:09:25.494723
(1 row)
```

SYS_SERVERLESS_USAGE

Utilisez SYS_SERVERLESS_USAGE pour afficher les détails de l'utilisation des ressources par Amazon Redshift sans serveur. Cette vue système ne s'applique pas aux clusters Amazon Redshift provisionnés.

Cette vue contient le récapitulatif de l'utilisation sans serveur, y compris la quantité de capacité de calcul utilisée pour traiter les requêtes et la quantité de stockage géré Amazon Redshift utilisée à une granularité de 1 minute. La capacité de calcul est mesurée en unités de traitement Redshift (RPUs) et mesurée pour les charges de travail que vous exécutez en secondes RPU par seconde. RPUs sont utilisés pour traiter des requêtes sur les données chargées dans l'entrepôt de données, demandées depuis un lac de données Amazon S3 ou accessibles depuis des bases de données opérationnelles à l'aide d'une requête fédérée. Amazon Redshift sans serveur conserve les informations dans SYS_SERVERLESS_USAGE pendant 7 jours.

Pour des exemples sur la facturation des frais de calcul, voir Facturation pour Amazon Redshift sans serveur.

SYS_SERVERLESS_USAGE n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
start_time	timestamp	Heure à laquelle l'intervalle a commencé.
end_time	timestamp	Heure à laquelle l'intervalle s'est terminé.
compute_secondes	double precision	Les secondes d'unité de calcul cumulée (RPU) consommée s pendant cet intervalle de temps. Cette valeur représent e la capacité RPU de base allouée au compte.

Nom de la colonne	Type de données	Description
compute_capacity	double precision	Nombre moyen d'unités de calcul (unités de traitement Redshift, ou RPUs) allouées pendant cet intervalle de temps. La valeur compute_capacity peut être modifiée dynamique ment.
data_storage	entier	Espace de stockage de données moyen en Mo utilisé pendant cet intervalle. Le stockage de données utilisé peut changer dynamiquement lorsque les données sont chargées ou supprimées de la base de données.
cross_region_transferred_data	entier	Les données cumulées transférées pour le partage de données entre régions, en octets, pendant cet intervalle de temps.

Nom de la colonne	Type de données	Description
charged_seconds	entier	Les secondes d'unité de calcul cumulée (RPU) facturées pendant cet intervalle de temps. Elles sont calculées après la fin des transactions et peuvent donc être égales à 0 pendant l'exécution d'une transaction. Utilisez charged_seconds pour calculer le coût d'un groupe de travail Amazon Redshift sans serveur. Cette valeur représente la capacité RPU allouée au groupe de travail Amazon Redshift sans serveur.

Notes d'utilisation

- Il existe des situations où compute_seconds est égal à 0 mais où charged_seconds est supérieur à 0, ou vice versa. Il s'agit d'un comportement normal résultant de la manière dont les données sont enregistrées dans la vue système. Pour une représentation plus précise des détails de l'utilisation sans serveur, nous vous recommandons d'agréger les données.

exemple

Pour obtenir le montant total des frais pour les heures RPU utilisées pendant un intervalle de temps en interrogeant charged_seconds, exécutez la requête suivante :

```
select trunc(start_time) "Day",
(sum(charged_seconds)/3600::double precision) * <Price for 1 RPU> as cost_incurred
from sys_serverless_usage
group by 1
order by 1
```

Notez qu'il peut y avoir un temps d'inactivité pendant l'intervalle. Le temps d'inactivité n'augmente pas la RPUs consommation.

SYS_SESSION_HISTORY

Utilisez SYS_SESSION_HISTORY pour afficher des informations sur les sessions actives en cours et l'historique des sessions.

SYS_SESSION_HISTORY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	char(50)	Identificateur de l'utilisateur qui a généré l'entrée.
session_id	entier	Identifiant de la session associée à l'instruction.
database_name	char(50)	Nom de la base de données.
status	char	Statut de la session. Les valeurs possibles sont `active`, `timed out` et `closed`.
session_timeout	entier	La durée maximale d'inactivité d'une séance avant son expiration (exprimée en secondes). 0 indique qu'aucun délai d'expiration n'a été défini.
start_time	timestamp	Horodatage selon lequel la connexion a été établie.
end_time	timestamp	Horodatage auquel la connexion s'est arrêtée.

exemple

Voici un exemple de sortie de SYS_SESSION_HISTORY.

```
select * from sys_session_history;
```

```
 user_id | session_id |   database_name  | status | session_timeout |
 start_time         |          end_time
---------+------------+------------------+--------+-----------------
+---------------------------+---------------------------
       1 | 1073971370 | dev              | closed |        0        | 2023-07-17
 15:50:12.030104 | 2023-07-17 15:50:12.123218
       1 | 1073979694 | dev              | closed |        0        | 2023-07-17
 15:50:24.117947 | 2023-07-17 15:50:24.131859
       1 | 1073873049 | dev              | closed |        0        | 2023-07-17
 15:49:29.067398 | 2023-07-17 15:49:29.070294
       1 | 1073873086 | database18127a4a | closed |        0        | 2023-07-17
 15:49:29.119018 | 2023-07-17 15:49:29.125925
       1 | 1073832112 | dev              | closed |        0        | 2023-07-17
 15:49:29.164688 | 2023-07-17 15:49:29.179631
       1 | 1073987697 | dev              | closed |        0        | 2023-07-17
 15:49:29.26749  | 2023-07-17 15:49:29.273034
       1 | 1073922429 | dev              | closed |        0        | 2023-07-17
 15:49:33.35315  | 2023-07-17 15:49:33.367499
       1 | 1073766783 | dev              | closed |        0        | 2023-07-17
 15:49:45.38237  | 2023-07-17 15:49:45.396902
       1 | 1073807506 | dev              | active |        0        | 2023-07-17
 15:51:48        |
```

SYS_SPATIAL_SIMPLIFY

Vous pouvez interroger la vue système SYS_SPATIAL_SIMPLIFY pour obtenir des informations sur les objets de géométrie spatiale simplifiée à l'aide de la commande COPY. Lorsque vous utilisez COPY sur un fichier de formes, vous pouvez spécifier les options d'ingestion SIMPLIFY `tolerance`, SIMPLIFY AUTO et SIMPLIFY AUTO `max_tolerance`. Le résultat de la simplification est résumé dans la vue système SYS_SPATIAL_SIMPLIFY.

Lorsque SIMPLIFY AUTO `max_tolerance` est défini, cette vue contient une ligne pour chaque géométrie ayant dépassé la taille maximale. Lorsque SIMPLIFY `tolerance` est défini, une ligne pour l'ensemble de l'opération COPY est stockée. Cette ligne fait référence à l'ID de la requête COPY et à la tolérance de simplification spécifiée.

Pour en savoir plus sur le chargement d'un fichier de forme (shapefile), consultez Chargement d'un shapefile dans Amazon Redshift.

SYS_SPATIAL_SIMPLIFY est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
query_id	bigint	L'ID de la requête (commande COPY) qui a généré cette ligne.
line_number	bigint	Lorsque l'option `SIMPLIFY AUTO` COPY est spécifiée, cette valeur est le numéro de l'enregistrement simplifié dans le fichier de formes.
maximum_tolerance	double precision	La valeur de tolérance de distance spécifiée dans la commande COPY. Il s'agit soit de la valeur de tolérance maximale via l'option `SIMPLIFY AUTO`, soit de la valeur de tolérance fixe via l'option `SIMPLIFY`.
initial_size	bigint	La taille en octets de la valeur de données `GEOMETRY` avant simplification.
simplified	char(1)	Lorsque l'option `SIMPLIFY AUTO` COPY est spécifiée, la valeur indique `t` si la géométrie a été simplifiée ou `f` si ce n'est pas le cas. La géométrie peut ne pas être simplifiée si, après la simplification avec la tolérance maximale donnée, la taille est toujours supérieure à la taille maximale de la géométrie.
final_size	bigint	Lorsque l'option `SIMPLIFY AUTO` COPY est spécifiée, il s'agit de la taille en octets de la géométrie après simplification.
final_tolerance	double precision	Tolérance finale choisie pour la simplification.

Exemple de requête

La requête suivante renvoie la liste des enregistrements simplifiés par COPY.

```
SELECT * FROM sys_spatial_simplify;


 query_id | line_number | maximum_tolerance | initial_size | simplified | final_size |
  final_tolerance
----------+-------------+-------------------+--------------+------------+------------
+----------------------
    20    |     1184704 |                -1 |      1513736 | t          |    1008808 |
  1.276386653895e-05
    20    |     1664115 |                -1 |      1233456 | t          |    1023584 |
 6.11707814796635e-06
```

SYS_STREAM_SCAN_ERRORS

Enregistre les erreurs relatives aux enregistrements chargés via l'ingestion en streaming.

SYS_STREAM_SCAN_ERRORS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
external_schema_name	character (128)	Nom du flux Kinesis ou du schéma de la rubrique Amazon MSK. Il est sensible à la casse.
stream_name	character (255)	Nom du flux ou de la rubrique. Il est sensible à la casse.
mv_name	character (128)	Nom de la vue matérialisée associée. Vide si aucune n'est définie. Il est sensible à la casse.
transaction_id	bigint	ID de transaction.
query_id	bigint	ID de requête.

Nom de la colonne	Type de données	Description
stream_timestamp_type	character (1)	Type d’horodatage du flux. Il est sensible à la casse.
stream_timestamp	horodatage sans fuseau horaire	Heure à laquelle l’enregistrement est arrivé.
record_time	horodatage sans fuseau horaire	Heure à laquelle le message d’erreur a été journalisé.
partition_id	character (128)	L' partition/shard identifiant. Il est sensible à la casse.
position	character (128)	Position de l’enregistrement. Elle correspond au numéro de séquence dans Kinesis ou au décalage dans Amazon MSK. Il est sensible à la casse.
error_code	entier	Code de l’erreur.
error_reason	character (128)	Raison de l’erreur. Il est sensible à la casse.

SYS_STREAM_SCAN_STATES

Enregistre les états d’analyse des enregistrements chargés via l’ingestion en streaming.

SYS_STREAM_SCAN_STATES est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
external_schema_name	character (128)	Nom du schéma externe. Il est sensible à la casse.
stream_name	character (255)	Nom du flux. Il est sensible à la casse.
mv_name	character (128)	Nom de la vue matérialisée associée. Vide si aucune n'est définie. Il est sensible à la casse.
transaction_id	bigint	ID de transaction.
query_id	bigint	ID de requête.
record_time	horodatage sans fuseau horaire	Heure à laquelle les données ont été journalisées.
partition_id	character (128)	ID de la partition. Il est sensible à la casse.
latest_position	character (128)	Position du dernier enregistrement lu dans le lot. Elle correspond au numéro de séquence dans Kinesis ou au décalage dans Amazon MSK. Il est sensible à la casse.
scanned_rows	bigint	Nombre d'enregistrements analysés dans le lot.
skipped_rows	bigint	Nombre d'enregistrements ignorés dans le lot.

Nom de la colonne	Type de données	Description
scanned_bytes	bigint	Nombre d'octets analysés dans le lot.
stream_record_time_min	horodatage sans fuseau horaire	Heure d'arrivée Kinesis ou Amazon MSK pour le premier enregistrement du lot.
stream_record_time_max	horodatage sans fuseau horaire	Heure d'arrivée Kinesis ou Amazon MSK pour le dernier enregistrement du lot.

La requête suivante affiche les données de flux et de rubrique pour des requêtes spécifiques.

```
select
 query_id,mv_name::varchar,external_schema_name::varchar,stream_name::varchar,sum(scanned_rows)
 total_records,
sum(scanned_bytes) total_bytes from sys_stream_scan_states where query in
 (5401180,8601939) group by 1,2,3,4;

  query_id  |    mv_name     | external_schema_name |   stream_name   | total_records |
  total_bytes
------------+----------------+----------------------+-----------------+---------------
+----------------
 5401180    | kinesistest    | kinesis              | kinesisstream   |    1493255696 |
 3209006490704
 8601939    | msktest        | msk                  | mskstream       |      14677023 |
   31056580668
(2 rows)
```

SYS_TRANSACTION_HISTORY

Utilisez SYS_TRANSACTION_HISTORY pour voir les détails d'une transaction lors du suivi d'une requête.

SYS_TRANSACTION_HISTORY n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	ID de l'utilisateur qui a généré l'entrée.
transaction_id	bigint	ID de la transaction.
isolation_level	text	Niveau d'isolation de la transaction. Les valeurs possibles sont `Serializable` et `Snapshot Isolation`.
État	text	Statut de la transaction. Les statuts possibles sont `committed` et `rolledback`.
transaction_start_time	timestamp	Heure de début de la transaction.
commit_start_time	timestamp	Heure de début de la validation.
commit_end_time	timestamp	Heure de fin de la validation.
blocks_committed	bigint	Nombre de blocs qui devaient être écrits dans le cadre de cette validation.
undo_transaction_id	bigint	ID de la transaction d'annulation si des transactions ont été annulées. Sinon, cette valeur est -1.

Exemples de requêtes

```
select * from sys_transaction_history order by transaction_start_time desc;

 user_id | transaction_id | isolation_level |   status   |   transaction_start_time
   |     commit_start_time      |      commit_end_time       | blocks_committed |
 undo_transaction_id
---------+----------------+-----------------+------------+--------------------------
+---------------------------+---------------------------+------------------
+---------------------
     100 |           1310 | Serializable    | committed  | 2023-08-27 21:03:11.822205 |
 2023-08-28 21:03:11.825287 | 2023-08-28 21:03:11.854883 |               17 |
       -1
     101 |           1345 | Serializable    | committed  | 2023-08-27 21:03:12.000278 |
 2023-08-28 21:03:12.003736 | 2023-08-28 21:03:12.030061 |               17 |
       -1
     102 |           1367 | Serializable    | committed  | 2023-08-27 21:03:12.1532   |
 2023-08-28 21:03:12.156124 | 2023-08-28 21:03:12.186468 |               17 |
       -1
     100 |           1370 | Serializable    | committed  | 2023-08-27 21:03:12.199316 |
 2023-08-28 21:03:12.204854 | 2023-08-28 21:03:12.238186 |               24 |
       -1
     100 |           1408 | Serializable    | committed  | 2023-08-27 21:03:53.891107 |
 2023-08-28 21:03:53.894825 | 2023-08-28 21:03:53.927465 |               17 |
       -1
     100 |           1409 | Serializable    | rolledback | 2023-08-27 21:03:53.936431 |
 2000-01-01 00:00:00        | 2023-08-28 21:04:08.712532 |                0 |
      1409
     101 |           1415 | Serializable    | committed  | 2023-08-27 21:04:24.283188 |
 2023-08-28 21:04:24.289196 | 2023-08-28 21:04:24.374318 |               25 |
       -1
     101 |           1416 | Serializable    | committed  | 2023-08-27 21:04:24.38818  |
 2023-08-28 21:04:24.391688 | 2023-08-28 21:04:24.415135 |               17 |
       -1
     100 |           1417 | Serializable    | rolledback | 2023-08-27 21:04:24.424252 |
 2000-01-01 00:00:00        | 2023-08-28 21:04:28.354826 |                0 |
      1417
     101 |           1418 | Serializable    | rolledback | 2023-08-27 21:04:24.425195 |
 2000-01-01 00:00:00        | 2023-08-28 21:04:28.680355 |                0 |
      1418
     100 |           1420 | Serializable    | committed  | 2023-08-27 21:04:28.697607 |
 2023-08-28 21:04:28.702374 | 2023-08-28 21:04:28.735541 |               23 |
       -1
```

```
    101 |           1421 | Serializable    | committed  | 2023-08-27 21:04:28.744854 |
2023-08-28 21:04:28.749344 | 2023-08-28 21:04:28.779029 |               23 |
       -1
    101 |           1423 | Serializable    | committed  | 2023-08-27 21:04:28.78942  |
2023-08-28 21:04:28.791556 | 2023-08-28 21:04:28.817485 |               16 |
       -1
    100 |           1430 | Serializable    | committed  | 2023-08-27 21:04:28.917788 |
2023-08-28 21:04:28.919993 | 2023-08-28 21:04:28.944812 |               16 |
       -1
    102 |           1494 | Serializable    | committed  | 2023-08-27 21:04:37.029058 |
2023-08-28 21:04:37.033137 | 2023-08-28 21:04:37.062001 |               16 |
       -1
```

SYS_UDF_LOG

Enregistre la génération des messages d'erreur et d'avertissement définis par le système au cours de l'exécution de la fonction définie par l'utilisateur (UDF).

SYS_UDF_LOG est visible uniquement par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
query_id	bigint	Identificateur de requête.
function_name	text	Nom de la fonction définie par l'utilisateur.
record_time	timestamp	Heure à laquelle l'enregistrement a été créé.
sequence	entier	Séquence d'un seul message de journal.
message	text	Texte du message du journal.

Exemples de requêtes

L'exemple suivant montre comment UDFs gérer les erreurs définies par le système. Le premier bloc présente la définition d'une fonction UDF qui renvoie l'inverse d'un argument. Lorsque vous exécutez la fonction et fournissez 0 comme argument, la fonction renvoie une erreur. La dernière instruction renvoie le message d'erreur journalisé dans SYS_UDF_LOG.

```
-- Create a function to find the inverse of a number.
CREATE OR REPLACE FUNCTION f_udf_inv(a int)

RETURNS float

IMMUTABLE AS $$return 1/a

$$ LANGUAGE plpythonu;

-- Run the function with 0 to create an error.
Select f_udf_inv(0);

-- Query SYS_UDF_LOG to view the message.
Select query_id, record_time, message::varchar from sys_udf_log;


query_id    |    record_time              |                message
----------+----------------------------
+-----------------------------------------------------
2211        | 2023-08-23 15:53:11.360538 |  ZeroDivisionError: integer division or
 modulo by zero line 2, in f_udf_inv\n return 1/a\n
```

L'exemple suivant ajoute un message de consignation et d'avertissement à la fonction UDF afin qu'une opération de division par zéro génère un message d'avertissement au lieu de s'arrêter avec un message d'erreur.

```
-- Create a function to find the inverse of a number and log a warning if you input 0.
CREATE OR REPLACE FUNCTION f_udf_inv_log(a int)
  RETURNS float IMMUTABLE
 AS $$
  import logging
  logger = logging.getLogger() #get root logger
  if a==0:
    logger.warning('You attempted to divide by zero.\nReturning zero instead of error.
\n')
```

```
    return 0
  else:
     return 1/a
$$ LANGUAGE plpythonu;

-- Run the function with 0 to trigger the warning.
Select f_udf_inv_log(0);

-- Query SYS_UDF_LOG to view the message.
Select query_id, record_time, message::varchar from sys_udf_log;

 query_id |        record_time         |                                        message
----------+----------------------------
+----------------------------------------------------------------------------------
     0   | 2023-08-23 16:10:48.833503 | WARNING: You attempted to divide by zero.
\nReturning zero instead of error.\n
```

SYS_UNLOAD_DETAIL

Utilisez SYS_UNLOAD_DETAIL pour afficher les détails d'une opération UNLOAD. L'opération enregistre une ligne pour chaque fichier créé par une instruction UNLOAD. Par exemple, si une instruction UNLOAD crée 12 fichiers, SYS_UNLOAD_DETAIL contiendra 12 lignes correspondantes.

SYS_UNLOAD_DETAIL est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	L'identificateur de l'utilisateur qui a généré l'entrée.
query_id	entier	Identifiant de requête de la commande UNLOAD.
session_id	entier	L'ID du processus associé à l'instruction de requête.

Nom de la colonne	Type de données	Description
transaction_id	bigint	ID de transaction associé à l'instruction de requête.
file_name	character(1280)	Chemin d'objet Amazon S3 complet du fichier.
start_time	timestamp	L'heure à laquelle la transacti on a commencé.
end_time	timestamp	L'heure à laquelle la transacti on s'est achevée.
line_count	bigint	Nombre de lignes déchargées dans le fichier.
transfer_size	bigint	Nombre d'octets transférés.
file_format	character(10)	Le format des fichiers déchargés.

Exemples de requêtes

La requête suivante affiche les détails de la requête déchargée, notamment le format, les lignes et le nombre de fichiers de la commande UNLOAD.

```
select query_id, substring(file_name, 0, 50), transfer_size, file_format from
 sys_unload_detail;
```

Exemple de sortie.

```
 query_id |                    substring                               | transfer_size
 | file_format
----------+------------------------------------------------------------
+---------------+-------------
     9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0000_part_00.gz  |
 395886 | Text
```

```
    9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0001_part_00.gz  |
 406444 | Text
    9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0002_part_00.gz  |
 409431 | Text
    9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0003_part_00.gz  |
 403051 | Text
    9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0004_part_00.gz  |
 413592 | Text
    9272 | s3://amzn-s3-demo-bucket/my_unload_doc_venue0005_part_00.gz  |
 395689 | Text
(6 rows)
```

SYS_UNLOAD_HISTORY

Utilisez SYS_UNLOAD_HISTORY pour afficher les détails des commandes UNLOAD. Chaque ligne représente une commande UNLOAD avec des statistiques accumulées pour certains champs. Elle contient les commandes UNLOAD en cours d'exécution et terminées.

SYS_UNLOAD_HISTORY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identifiant de l'utilisateur qui a envoyé le déchargement.
query_id	bigint	Identifiant de requête de la commande UNLOAD.
transaction_id	bigint	Identificateur de transaction.
session_id	entier	Identifiant de processus du processus exécutant le déchargement.
database_name	text	Nom de la base de données à laquelle l'utilisateur était

Nom de la colonne	Type de données	Description
		connecté lorsque l'opération a été émise.
État	text	Statut de la commande UNLOAD. Les valeurs valides sont les suivantes : `running`, `completed` , `aborted` et `unknown`.
start_time	timestamp	Heure à laquelle le déchargement a commencé.
end_time	timestamp	Heure à laquelle le déchargement s'est terminé.
duration	bigint	Durée (microsecondes) passée dans la commande UNLOAD.
file_format	text	Format de fichier des fichiers de sortie.
compression_type	text	Type de compression.
unloaded_location	text	Emplacement Amazon S3 des fichiers déchargés.
unloaded_rows	bigint	Nombre de lignes.
unloaded_files_count	bigint	Nombre de fichiers du fichier de sortie.
unloaded_files_size	bigint	Taille de fichier du fichier de sortie.
error_message	text	Message d'erreur de la commande UNLOAD.

Exemples de requêtes

La requête suivante affiche les détails de la requête déchargée, notamment le format, les lignes et le nombre de fichiers de la commande UNLOAD.

```
SELECT query_id,
       file_format,
       start_time,
       duration,
       unloaded_rows,
       unloaded_files_count
FROM sys_unload_history
ORDER BY query_id,
file_format limit 100;
```

Exemple de sortie.

```
 query_id | file_format |         start_time         | duration | unloaded_rows |
 unloaded_files_count
----------+-------------+----------------------------+----------+---------------
+----------------------
   527067 | Text        | 2022-02-09 05:18:35.844452 |  5932478 |            10 |
             1
```

SYS_USERLOG

Enregistre les détails des modifications suivantes apportées à un utilisateur de base de données :

- Créer un utilisateur
- Supprimer un utilisateur
- Modifier un utilisateur (renommer)
- Modifier un utilisateur (modifier les propriétés)

Vous pouvez interroger cette vue pour obtenir des informations sur les groupes de travail sans serveur et les clusters provisionnés.

SYS_USERLOG n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	Identifiant de l'utilisateur qui a envoyé le déchargement.
user_name	character(50)	Nom d'utilisateur de l'utilisa teur affecté par la modification.
original_user_name	character(50)	Nom d'utilisateur d'origine dans une action de changement de nom. Ce champ est vide pour toutes les autres actions.
action	character(10)	Action qui s'est produite. Les valeurs valides sont alter (modification), create (création), drop (suppression) et rename (changement de nom).
has_create_db_privs	entier	Si c'est vrai (valeur 1), indique que l'utilisateur dispose d'autorisations de création de base de données.
is_superuser	entier	Si c'est vrai (valeur 1), indique que l'utilisateur peut mettre à jour les catalogues système.
has_update_catalog_privs	entier	Si c'est vrai (valeur 1), indique que l'utilisateur peut mettre à jour les catalogues système.
password_expiration	timestamp	Date d'expiration du mot de passe.

Nom de la colonne	Type de données	Description
session_id	entier	ID du processus.
transaction_id	bigint	ID de transaction.
record_time	timestamp	Heure au format UTC à laquelle la requête a démarré.

Exemples de requêtes

L'exemple suivant effectue quatre actions utilisateur, puis interroge la vue SYS_USERLOG.

```
CREATE USER userlog1 password 'Userlog1';
ALTER USER userlog1 createdb createuser;
ALTER USER userlog1 rename  to userlog2;
DROP user userlog2;

SELECT user_id, user_name, original_user_name, action, has_create_db_privs,
 is_superuser from SYS_USERLOG order by record_time desc;
```

```
user_id |  user_name | original_user_name |  action | has_create_db_privs |
 is_superuser
--------+------------+--------------------+---------+---------------------+------------
    108 |  userlog2  |                    | drop    |                   1 |   1
    108 |  userlog2  |     userlog1       | rename  |                   1 |   1
    108 |  userlog1  |                    | alter   |                   1 |   1
    108 |  userlog1  |                    | create  |                   0 |   0
 (4 rows)
```

SYS_VACUUM_HISTORY

Utilisez SYS_VACUUM_HISTORY pour afficher les détails concernant les requêtes de vidage (VACUUM). Pour en savoir plus sur la commande VACUUM, consultez VACUUM.

SYS_VACUUM_HISTORY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
user_id	entier	ID de l'utilisateur qui a lancé la requête.
transaction_id	long	ID de transaction de l'instruction VACUUM.
query_id	long	Identifiant de requête pour l'instruction VACUUM. Vous pouvez joindre cette table à la vue SYS_QUERY_DETAIL pour afficher chacune des instructions SQL qui ont été exécutées pour une transaction VACUUM donnée. Si vous exécutez cette opération sur l'ensemble de la base de données, chaque table est aspirée dans une transaction distincte. Pour les opérations VACUUM automatisées, cette valeur est null.
database_name	text	Nom de la base de données.
nom_schéma	text	Nom du schéma.
table_name	text	Nom de la table.
table_id	entier	ID de la table.
vacuum_type	character	Type de l'opération VACUUM. Les valeurs possibles sont les suivantes : • **`Delete`**

Nom de la colonne	Type de données	Description
		• **`Sort`** • **`Reindex`** • **`Recluster`** • **`Full`** Pour en savoir plus sur les types d'opération VACUUM, consultez VACUUM.
is_automatic	boolean	`true` s'il s'agit d'une opération VACUUM automatique. Sinon la valeur est renvoy, `false`.
status	character	Description de l'activité en cours exécutée dans le cadre de l'opération VACUUM : • Initialiser • Tri • Fusionner • Suppression • Select • Échec • Complet • Ignoré • Génération de l'ordre INTERLEAVED SORTKEY
start_time	timestamp	Heure à laquelle l'opération VACUUM a commencé.

Nom de la colonne	Type de données	Description
end_time	timestamp	Heure à laquelle l'opération VACUUM a pris fin. Si l'opération est en cours, ce champ est vide.
record_time	timestamp	Heure à laquelle l'opération VACUUM a été enregistrée dans SYS_VACUUM_HISTORY.
duration	entier	Nombre de microsecondes entre le début et la fin de l'opération VACUUM. Si l'opération VACUUM est en cours, ce champ est vide.
rows_before_vacuum	bigint	Nombre réel de lignes de la table, plus les lignes supprimées qui sont toujours stockées sur disque (en attente d'une opération VACUUM).
size_before_vacuum	entier	Taille de la table avant le début de l'opération VACUUM, en Mo.
reclaimable_rows	bigint	Nombre de lignes que l'opération VACUUM estime pouvoir récupérer avant de démarrer.
reclaimed_rows	bigint	Nombre de lignes récupérées par l'opération VACUUM.
reclaimed_blocks	bigint	Nombre de blocs récupérés par l'opération VACUUM.

Nom de la colonne	Type de données	Description
sortedrows_before_vacuum	entier	Nombre de lignes triées dans la table avant le début de l'opération VACUUM.
sortedrows_after_vacuum	entier	Nombre supplémentaire de lignes triées dans la table une fois l'opération VACUUM terminée. Ce nombre n'inclut pas les lignes comptées dans `sortedrows_before_vacuum` .

Cartographie des vues système pour la migration vers les vues de surveillance SYS

Lorsque vous migrez votre cluster provisionné Amazon Redshift vers Amazon Redshift sans serveur, vos requêtes de surveillance ou de diagnostic peuvent faire référence à des vues du système qui ne sont disponibles que sur les clusters provisionnés. Vous pouvez mettre à jour vos requêtes pour utiliser les vues de surveillance SYS. Cette page fournit des mappages de vues provisionnés uniquement vers SYS afin que vous puissiez vous y référer lors de la mise à jour de vos requêtes.

Rubriques

- SYS_QUERY_HISTORY
- SYS_QUERY_DETAIL
- SYS_RESTORE_LOG
- SYS_RESTORE_STATE
- SYS_TRANSACTION_HISTORY
- SYS_QUERY_TEXT
- SYS_CONNECTION_LOG
- SYS_SESSION_HISTORY
- SYS_LOAD_DETAIL
- SYS_LOAD_HISTORY

- SYS_LOAD_ERROR_DETAIL
- SYS_UNLOAD_HISTORY
- SYS_UNLOAD_DETAIL
- SYS_COPY_REPLACEMENTS
- SYS_DATASHARE_USAGE_CONSUMER
- SYS_DATASHARE_USAGE_PRODUCER
- SYS_DATASHARE_CROSS_REGION_USAGE
- SYS_DATASHARE_CHANGE_LOG
- SYS_EXTERNAL_QUERY_DETAIL
- SYS_EXTERNAL_QUERY_ERROR
- SYS_VACUUM_HISTORY
- SYS_ANALYZE_HISTORY
- SYS_ANALYZE_COMPRESSION_HISTORY
- SYS_MV_REFRESH_HISTORY
- SYS_MV_STATE
- SYS_PROCEDURE_CALL
- SYS_PROCEDURE_MESSAGES
- SYS_UDF_LOG
- SYS_USERLOG
- SYS_SCHEMA_QUOTA_VIOLATIONS
- SYS_SPATIAL_SIMPLIFY

SYS_QUERY_HISTORY

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_QUERY_HISTORY.

- STL_DDLTEXT
- STL_ERROR
- STL_QUERY

- STL_UTILITYTEXT
- STL_WLM_QUERY
- STV_INFLIGHT
- STV_RECENTS
- STV_WLM_QUERY_STATE
- SVL_COMPILE
- SVL_MULTI_STATEMENT_VIOLATIONS
- SVL_QLOG
- SVL_QUERY_QUEUE_INFO
- SVL_STATEMENTTEXT
- SVL_TERMINATE

SYS_QUERY_DETAIL

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_QUERY_DETAIL.

- STL_AGGR
- STL_ALERT_EVENT_LOG
- STL_BCAST
- STL_DELETE
- STL_DIST
- STL_EXPLAIN
- STL_HASH
- STL_HASHJOIN
- STL_INSERT
- STL_LIMIT
- STL_MERGE
- STL_MERGEJOIN
- STL_NESTLOOP
- STL_PARSE

- STL_PLAN_INFO
- STL_PROJECT
- STL_QUERY_METRICS
- STL_RETURN
- STL_SAVE
- STL_SCAN
- STL_SORT
- STL_STREAM_SEGS
- STL_UNIQUE
- STL_WINDOW
- STV_EXEC_STATE
- STV_QUERY_METRICS
- SVCS_QUERY_SUMMARY
- SVL_QUERY_METRICS
- SVL_QUERY_METRICS_SUMMARY
- SVL_QUERY_REPORT
- SVL_QUERY_SUMMARY
- SVV_QUERY_STATE

SYS_RESTORE_LOG

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_RESTORE_LOG.

- SVL_RESTORE_ALTER_TABLE_PROGRESS

SYS_RESTORE_STATE

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_RESTORE_STATE.

- STV_XRESTORE_ALTER_QUEUE_STATE

SYS_TRANSACTION_HISTORY

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_TRANSACTION_HISTORY.

- STL_COMMIT_STATS
- STL_TR_CONFLICT
- STL_UNDONE

SYS_QUERY_TEXT

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_QUERY_TEXT.

- STL_QUERYTEXT

SYS_CONNECTION_LOG

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_CONNECTION_LOG.

- STL_CONNECTION_LOG

SYS_SESSION_HISTORY

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_SESSION_HISTORY.

- STL_SESSIONS
- STL_RESTARTED_SESSIONS
- STV_SESSIONS

SYS_LOAD_DETAIL

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_LOAD_DETAIL.

- STL_LOAD_COMMITS

SYS_LOAD_HISTORY

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_LOAD_HISTORY.

- STL_LOAD_COMMITS

SYS_LOAD_ERROR_DETAIL

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_LOAD_ERROR_DETAIL.

- STL_LOADERROR_DETAIL
- STL_LOAD_ERRORS

SYS_UNLOAD_HISTORY

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_UNLOAD_HISTORY.

- STL_UNLOAD_LOG

SYS_UNLOAD_DETAIL

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_UNLOAD_DETAIL.

- STL_UNLOAD_LOG

SYS_COPY_REPLACEMENTS

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_COPY_REPLACEMENTS.

- STL_REPLACEMENTS

SYS_DATASHARE_USAGE_CONSUMER

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_DATASHARE_USAGE_CONSUMER.

- SVL_DATASHARE_USAGE_CONSUMER

SYS_DATASHARE_USAGE_PRODUCER

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_DATASHARE_USAGE_PRODUCER.

- SVL_DATASHARE_USAGE_PRODUCER

SYS_DATASHARE_CROSS_REGION_USAGE

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_DATASHARE_CROSS_REGION_USAGE.

- SVL_DATASHARE_CROSS_REGION_USAGE

SYS_DATASHARE_CHANGE_LOG

Une partie ou la totalité des colonnes de la table suivante est également définie dans SYS_DATASHARE_CHANGE_LOG.

- SVL_DATASHARE_CHANGE_LOG

SYS_EXTERNAL_QUERY_DETAIL

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_EXTERNAL_QUERY_DETAIL.

- SVL_FEDERATED_QUERY
- SVL_S3LIST
- SVL_S3QUERY
- SVL_S3QUERY_SUMMARY

SYS_EXTERNAL_QUERY_ERROR

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_EXTERNAL_QUERY_ERROR.

- SVL_SPECTRUM_SCAN_ERROR

SYS_VACUUM_HISTORY

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_VACUUM_HISTORY.

- STL_VACUUM
- SVL_VACUUM_PERCENTAGE
- SVV_VACUUM_PROGRESS
- SVV_VACUUM_SUMMARY

SYS_ANALYZE_HISTORY

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_ANALYZE_HISTORY.

- STL_ANALYZE

SYS_ANALYZE_COMPRESSION_HISTORY

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_ANALYZE_COMPRESSION_HISTORY.

- STL_ANALYZE_COMPRESSION

SYS_MV_REFRESH_HISTORY

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_MV_REFRESH_HISTORY.

- SVL_MV_REFRESH_STATUS

SYS_MV_STATE

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_MV_STATE.

- STL_MV_STATE

SYS_PROCEDURE_CALL

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_PROCEDURE_CALL.

- SVL_STORED_PROC_CALL

SYS_PROCEDURE_MESSAGES

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_PROCEDURE_MESSAGES.

- SVL_STORED_PROC_MESSAGES

SYS_UDF_LOG

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_UDF_LOG.

- SVL_UDF_LOG

SYS_USERLOG

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_USERLOG.

- STL_USERLOG

SYS_SCHEMA_QUOTA_VIOLATIONS

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_SCHEMA_QUOTA_VIOLATIONS.

- STL_SCHEMA_QUOTA_VIOLATIONS

SYS_SPATIAL_SIMPLIFY

Une partie ou la totalité des colonnes des tables suivantes est également définie dans SYS_SPATIAL_SIMPLIFY.

- SVL_SPATIAL_SIMPLIFICATION

Surveillance système (clusters mis en service uniquement)

Les tables et les vues système suivantes peuvent être interrogées sur les clusters mis en service. Les tables et les vues système STL et STV contiennent un sous-ensemble des données qui se trouvent dans plusieurs tables système. Elles permettent d'accéder de façon simple et rapide aux données fréquemment interrogées qui se trouvent dans ces tables.

Les vues SVCS fournissent des détails sur les requêtes effectuées sur les clusters principaux et les clusters de mise à l'échelle de simultanéité. Les vues SVL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal, à l'exception de SVL_STATEMENTTEXT. SVL_STATEMENTTEXT peut contenir des informations pour les requêtes exécutées sur des clusters de dimensionnement simultané ainsi que sur le cluster principal.

Rubriques

- Vues STL pour la journalisation
- Tables STV pour les données d’instantanés
- Vues SVCS pour le cluster principal et les clusters de mise à l’échelle de la simultanéité
- Vues SVL pour le cluster principal

Vues STL pour la journalisation

Les vues système STL sont générées à partir des fichiers journaux Amazon Redshift afin de fournir un historique du système.

Ces fichiers résident sur chaque nœud du cluster de l'entrepôt des données. Les vues STL prennent les informations des journaux et les mettent sous forme de vues utilisables par les administrateurs système.

Conservation des journaux : les vues du système STL conservent sept jours d'historique des journaux. La conservation des journaux est garantie pour toutes les tailles de clusters et tous les types de nœuds, et elle n'est pas affectée par les modifications de la charge de travail du cluster. La conservation des journaux n'est pas non plus affectée par le statut du cluster, par exemple lorsque le cluster est suspendu. Vous disposez d'un historique des journaux de moins de sept jours uniquement dans le cas où le cluster est nouveau. Vous n'avez aucune action à effectuer pour conserver les journaux, mais vous devez régulièrement copier les données des journaux dans d'autres tables ou les décharger sur Amazon S3 pour conserver les données des journaux datant de plus de 7 jours.

Rubriques

- STL_AGGR
- STL_ALERT_EVENT_LOG
- STL_ANALYZE
- STL_ANALYZE_COMPRESSION
- STL_BCAST
- STL_COMMIT_STATS
- STL_CONNECTION_LOG
- STL_DDLTEXT
- STL_DELETE
- STL_DISK_FULL_DIAG
- STL_DIST
- STL_ERROR
- STL_EXPLAIN
- STL_FILE_SCAN
- STL_HASH
- STL_HASHJOIN
- STL_INSERT
- STL_LIMIT
- STL_LOAD_COMMITS

- STL_LOAD_ERRORS
- STL_LOADERROR_DETAIL
- STL_MERGE
- STL_MERGEJOIN
- STL_MV_STATE
- STL_NESTLOOP
- STL_PARSE
- STL_PLAN_INFO
- STL_PROJECT
- STL_QUERY
- STL_QUERY_METRICS
- STL_QUERYTEXT
- STL_REPLACEMENTS
- STL_RESTARTED_SESSIONS
- STL_RETURN
- STL_S3CLIENT
- STL_S3CLIENT_ERROR
- STL_SAVE
- STL_SCAN
- STL_SCHEMA_QUOTA_VIOLATIONS
- STL_SESSIONS
- STL_SORT
- STL_SSHCLIENT_ERROR
- STL_STREAM_SEGS
- STL_TR_CONFLICT
- STL_UNDONE
- STL_UNIQUE
- STL_UNLOAD_LOG
- STL_USAGE_CONTROL
- STL_USERLOG

- STL_UTILITYTEXT
- STL_VACUUM
- STL_WINDOW
- STL_WLM_ERROR
- STL_WLM_RULE_ACTION
- STL_WLM_QUERY

STL_AGGR

Analyse les étapes d'exécution de l'agrégation pour les requêtes. Ces étapes se produisent lors de l'exécution des fonctions d'agrégation et des clauses GROUP BY.

STL_AGGR est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_AGGR contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.

Nom de la colonne	Type de données	Description
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
octets	bigint	Taille, en octets, de toutes les lignes de sortie de l'étape.
slots	entier	Nombre de compartiments de hachage.
occupied	entier	Nombre d'emplacements qui contiennent des enregistr ements.
maxlength	entier	Taille de l'emplacement le plus grand.
tbl	entier	ID de table.
is_diskba sed	character(1)	Si la valeur est true (t), la requête a été exécutée en tant qu'opération sur disque. Si elle est false (f), la requête a été exécutée en mémoire.

Nom de la colonne	Type de données	Description
workmem	bigint	Nombre d'octets de mémoire de travail assignés à l'étape.
type	character(6)	Type d'étape. Les valeurs valides sont : • HASHED. Indique que l'étape a utilisé l'agrégation groupée, non triée. • PLAIN. Indique que l'étape a utilisé l'agrégation non groupée, scalaire. • SORTED. Indique que l'étape a utilisé l'agrégation groupée, triée.
redimensionne	entier	Information à utilisation interne uniquement.
flushable	entier	Information à utilisation interne uniquement.

Exemples de requêtes

renvoie les informations sur les étapes de l'exécution de l'agrégation pour SLICE 1 et TBL 239.

```
select query, segment, bytes, slots, occupied, maxlength, is_diskbased, workmem, type
from stl_aggr where slice=1 and tbl=239
order by rows
limit 10;
```

```
 query | segment | bytes |  slots  | occupied | maxlength | is_diskbased |  workmem  |
 type
-------+---------+-------+---------+----------+-----------+--------------+-----------
+--------
   562 |       1 |     0 | 4194304 |        0 |         0 | f            | 383385600 |
 HASHED
   616 |       1 |     0 | 4194304 |        0 |         0 | f            | 383385600 |
 HASHED
   546 |       1 |     0 | 4194304 |        0 |         0 | f            | 383385600 |
 HASHED
```

```
  547 |       0 |     8 |       0 |       0 |       0 | f           |         0 |
PLAIN
  685 |       1 |    32 | 4194304 |       1 |       0 | f           | 383385600 |
HASHED
  652 |       0 |     8 |       0 |       0 |       0 | f           |         0 |
PLAIN
  680 |       0 |     8 |       0 |       0 |       0 | f           |         0 |
PLAIN
  658 |       0 |     8 |       0 |       0 |       0 | f           |         0 |
PLAIN
  686 |       0 |     8 |       0 |       0 |       0 | f           |         0 |
PLAIN
  695 |       1 |    32 | 4194304 |       1 |       0 | f           | 383385600 |
HASHED
(10 rows)
```

STL_ALERT_EVENT_LOG

Enregistre une alerte quand l'optimiseur de requête identifie les conditions qui peuvent indiquer des problèmes de performances. Utilisez la vue STL_ALERT_EVENT_LOG pour identifier les opportunités d'améliorer les performances des requêtes.

Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Pour plus d'informations, consultez Traitement des requêtes.

STL_ALERT_EVENT_LOG est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_ALERT_EVENT_LOG contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
pid	entier	ID de processus associé à l'instruction et à la tranche. La même requête peut en avoir plusieurs PIDs si elle s'exécute sur plusieurs tranches.
xid	bigint	ID de transaction associé à l'instruction.
event	character (1024)	Description de l'événement d'alerte.
solution	character (1024)	Solution recommandée.
event_time	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .

Notes d'utilisation

Vous pouvez utiliser STL_ALERT_EVENT_LOG pour identifier les problèmes potentiels de votre requête, puis suivre les pratiques décrites dans Réglage des performances des

requêtes afin d'optimiser la conception de votre base de données et de réécrire vos requêtes. STL_ALERT_EVENT_LOG enregistre les alertes suivantes :

- Statistiques manquantes

  Il manque des statistiques. Exécutez ANALYZE après les chargements de données ou les mises à jour importantes, et utilisez STATUPDATE avec les opérations COPY. Pour plus d'informations, consultez Bonnes pratiques Amazon Redshift pour la conception de requêtes.

- Boucle imbriquée

  Une boucle imbriquée est généralement un produit cartésien. Évaluez votre requête afin de vous assurer que toutes les tables participantes sont unies efficacement.

- Filtre très sélectif

  Le rapport entre le nombre de lignes retournées et le nombre de lignes analysées est inférieur à 0,05. Le nombre de lignes analysées est la valeur de `rows_pre_user_filter` et le nombre de lignes renvoyées est la valeur de lignes dans la vue système STL_SCAN. Indique que la requête analyse un nombre inhabituellement grand de lignes pour déterminer le jeu de résultats. La raison peut en être des clés de tri manquantes ou incorrectes. Pour plus d'informations, consultez Clés de tri.

- Lignes fantôme excessives

  Une analyse a ignoré un nombre relativement grand de lignes marquées comme supprimées, mais pas vidées, ou de lignes qui ont été insérées, mais pas validées. Pour plus d'informations, consultez Exécution de l'opération VACUUM sur les tables.

- Grande Distribution

  Plus de 1 000 000 de lignes ont été redistribuées pour une jointure par hachage ou une agrégation. Pour plus d'informations, consultez Distribution des données pour l'optimisation des requêtes.

- Large diffusion

  Plus de 1 000 000 de lignes ont été diffusées pour une jointure par hachage. Pour plus d'informations, consultez Distribution des données pour l'optimisation des requêtes.

- Exécution en série

  Un style de redistribution DS_DIST_ALL_INNER a été indiqué dans le plan de requête, ce qui force une exécution en série, car la totalité de la table interne a été redistribuée sur un seul nœud. Pour plus d'informations, consultez Distribution des données pour l'optimisation des requêtes.

Exemples de requêtes

La requête suivante affiche les événements d'alerte pour quatre requêtes.

```
SELECT query, substring(event,0,25) as event,
substring(solution,0,25) as solution,
trim(event_time) as event_time from stl_alert_event_log order by query;

 query |             event             |          solution            |     event_time

-------+-------------------------------+-------------------------------
+---------------------
  6567 | Missing query planner statist | Run the ANALYZE command      | 2014-01-03
 18:20:58
  7450 | Scanned a large number of del | Run the VACUUM command to rec| 2014-01-03
 21:19:31
  8406 | Nested Loop Join in the query | Review the join predicates to| 2014-01-04
 00:34:22
 29512 | Very selective query filter:r | Review the choice of sort key| 2014-01-06
 22:00:00

(4 rows)
```

STL_ANALYZE

Enregistre les détails des opérations ANALYSE.

STL_ANALYZE n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_ANALYZE_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.

Nom de la colonne	Type de données	Description
xid	long	ID de transaction.
database	char(30)	Nom de la base de données.
table_id	entier	ID de la table.
status	char(15)	Résultat de la commande d'analyse. Les valeurs possibles sont `Full`, `Skipped` et `PredicateColumn` .
rows	double	Nombre total de lignes de la table.
modified_rows	double	Nombre total de lignes qui ont été modifiées depuis la dernière opération ANALYZE.
threshold_percent	entier	Valeur du paramètre `analyze_threshold_percent` .
is_auto	char(1)	La valeur est true (`t`) si l'opération incluait une opération d'analyse Amazon Redshift par défaut. La valeur est false (`f`) si la commande ANALYZE a été exécutée explicitement.
starttime	timestamp	Heure UTC à laquelle l'opération ANALYZE a commencé à s'exécuter.
endtime	timestamp	Heure UTC à laquelle l'opération ANALYZE a fini de s'exécuter.
prevtime	timestamp	Heure UTC à laquelle la table a été analysée précédemment.
num_predicate_cols	entier	Nombre actuel de colonnes de prédicat dans la table.
num_new_predicate_cols	entier	Nombre de nouvelles colonnes de prédicat dans la table depuis l'opération ANALYZE précédente.

Nom de la colonne	Type de données	Description
is_background	character(1)	La valeur est true (t) si l'analyse a été exécutée par une opération d'analyse automatique. Sinon, la valeur est false (f).
auto_analyze_phase	character(100)	Réservé pour un usage interne.
nom_schéma	char(128)	Nom du schéma de la table.
table_name	char(136)	Nom de la table.

Exemples de requêtes

L'exemple suivant joint la table STV_TBL_PERM pour afficher les détails des noms et d'exécution de la table.

```
select distinct a.xid, trim(t.name) as name, a.status, a.rows, a.modified_rows,
 a.starttime, a.endtime
from stl_analyze a
join stv_tbl_perm t  on t.id=a.table_id
where name = 'users'
order by starttime;

xid    | name  | status          | rows  | modified_rows | starttime           |
 endtime
-------+-------+-----------------+-------+---------------+---------------------
+---------------------
  1582 | users | Full            | 49990 |         49990 | 2016-09-22 22:02:23 |
 2016-09-22 22:02:28
244287 | users | Full            | 24992 |         74988 | 2016-10-04 22:50:58 |
 2016-10-04 22:51:01
244712 | users | Full            | 49984 |         24992 | 2016-10-04 22:56:07 |
 2016-10-04 22:56:07
245071 | users | Skipped         | 49984 |             0 | 2016-10-04 22:58:17 |
 2016-10-04 22:58:17
245439 | users | Skipped         | 49984 |          1982 | 2016-10-04 23:00:13 |
 2016-10-04 23:00:13
```

```
(5 rows)
```

STL_ANALYZE_COMPRESSION

Enregistre les détails pour les opérations d'analyse de compression au cours de commandes COPY ou ANALYZE COMPRESSION.

STL_ANALYZE_COMPRESSION est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_ANALYZE_COMPRESSION_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
start_time	timestamp	Heure à laquelle l'opération d'analyse de compression a commencé.
xid	bigint	ID de transaction de l'opération d'analyse de compression.
tbl	entier	ID de la table analysée.
tablename	character(128)	Nom de la table analysée.
col	entier	Index de la colonne de la table analysée pour déterminer l'encodage de compression.
old_encoding	character(15)	Type d'encodage avant l'analyse de compression.

Nom de la colonne	Type de données	Description
new_encoding	character(15)	Type d'encodage après l'analyse de compression.
mode	character(14)	Les valeurs possibles sont : PRESET Spécifie que `new_encoding` est déterminé par la commande COPY Amazon Redshift en fonction du type de données de la colonne. Aucune donnée échantillonnée. ON Spécifie que `new_encoding` est déterminé par la commande COPY Amazon Redshift en fonction de l'analyse d'un exemple de données. ANALYZE ONLY Spécifie que `new_encoding` est déterminé par la commande ANALYZE COMPRESSION Amazon Redshift en fonction de l'analyse d'un exemple de données. Toutefois, le type d'encodage de la colonne analysée n'est pas modifié.
meilleur encodage par compression	character(15)	Type de codage offrant le meilleur taux de compression.
octets_recommandés	character(15)	Les octets utilisés en adoptant le nouveau codage.
best_compression_bytes	character(15)	Les octets utilisés en adoptant le meilleur codage de compression.

Nom de la colonne	Type de données	Description
ndv	bigint	Le nombre de valeurs distinctes dans les lignes échantill onnées.

Exemples de requêtes

L'exemple suivant examine les détails de l'analyse de compression effectuée sur la table `lineitem` par la dernière commande COPY exécutée dans la même séance.

```
select xid, tbl, btrim(tablename) as tablename, col, old_encoding, new_encoding,
 best_compression_encoding, mode
from stl_analyze_compression
where xid = (select xid from stl_query where query = pg_last_copy_id()) order by col;

 xid  |  tbl   | tablename | col |  old_encoding   |  new_encoding   |
 best_compression_encoding |      mode
------+--------+-----------+-----+-----------------+-----------------
+---------------------------+----------------
 5308 | 158961 | $lineitem |   0 | mostly32        | az64            | delta
          | ON
 5308 | 158961 | $lineitem |   1 | mostly32        | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   2 | lzo             | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   3 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   4 | bytedict        | az64            | bytedict
          | ON
 5308 | 158961 | $lineitem |   5 | mostly32        | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   6 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   7 | delta           | az64            | az64
          | ON
 5308 | 158961 | $lineitem |   8 | lzo             | lzo             | lzo
          | ON
 5308 | 158961 | $lineitem |   9 | runlength       | runlength       | runlength
          | ON
```

```
 5308 | 158961 | $lineitem |  10 | delta          | az64           | az64
          | ON
 5308 | 158961 | $lineitem |  11 | delta          | az64           | az64
          | ON
 5308 | 158961 | $lineitem |  12 | delta          | az64           | az64
          | ON
 5308 | 158961 | $lineitem |  13 | bytedict       | bytedict       | bytedict
          | ON
 5308 | 158961 | $lineitem |  14 | bytedict       | bytedict       | bytedict
          | ON
 5308 | 158961 | $lineitem |  15 | text255        | text255        | text255
          | ON
(16 rows)
```

STL_BCAST

Enregistre les informations sur l'activité réseau pendant l'exécution des étapes de requête qui diffusent des données. Le trafic réseau est capturé par le nombre de lignes, d'octets et de paquets envoyés sur le réseau pendant une étape donnée sur une tranche donnée. La durée de l'étape est la différence entre les heures de début et de fin enregistrées.

Pour identifier les étapes de diffusion d'une requête, recherchez les étiquettes bcast dans la vue SVL_QUERY_SUMMARY ou exécutez la commande EXPLAIN, puis cherchez les attributs de l'étape qui incluent bcast.

STL_BCAST est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_BCAST contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
octets	bigint	Taille, en octets, de toutes les lignes de sortie de l'étape.
paquets	entier	Nombre total de paquets envoyés sur le réseau.

Exemples de requêtes

L'exemple suivant renvoie les informations de diffusion pour les requêtes où il y a un ou plusieurs paquets, et où la différence entre le début et la fin de la requête est égal ou supérieur à une seconde.

```
select query, slice, step, rows, bytes, packets, datediff(seconds, starttime, endtime)
from stl_bcast
where packets>0 and datediff(seconds, starttime, endtime)>0;
```

```
 query | slice | step | rows | bytes | packets | date_diff
-------+-------+------+------+-------+---------+-----------
   453 |     2 |    5 |    1 |   264 |       1 |         1
   798 |     2 |    5 |    1 |   264 |       1 |         1
  1408 |     2 |    5 |    1 |   264 |       1 |         1
  2993 |     0 |    5 |    1 |   264 |       1 |         1
  5045 |     3 |    5 |    1 |   264 |       1 |         1
  8073 |     3 |    5 |    1 |   264 |       1 |         1
  8163 |     3 |    5 |    1 |   264 |       1 |         1
  9212 |     1 |    5 |    1 |   264 |       1 |         1
  9873 |     1 |    5 |    1 |   264 |       1 |         1
(9 rows)
```

STL_COMMIT_STATS

Fournit les métriques liées à la validation des performances, y compris la chronologie des différentes étapes de validation et le nombre de blocs validés. Requête STL_COMMIT_STATS pour déterminer quelle partie d'une transaction a été consacrée à la validation et quelle partie se trouve dans la file d'attente.

STL_COMMIT_STATS n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_TRANSACTION_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
xid	bigint	Transaction en cours de validation.
node	integer	Numéro du nœud. -1 est le nœud principal.
startqueue	timestamp	Début de la file d'attente pour la validation.
startwork	timestamp	Début de validation.
endflush	timestamp	Fin de la phase de vidage des blocs erronés.
endstage	timestamp	Fin de la phase intermédiaire des métadonnées.
endlocal	timestamp	Fin de la phase de validation locale.
startglobal	timestamp	Début de la phase globale.
endtime	timestamp	Fin de la validation.
queuelen	bigint	Nombre de transactions qui se trouvaient devant cette opération dans la file d'attente de validation.
permblocks	bigint	Nombre de blocs permanents existants au moment de la validation.
newblocks	bigint	Nombre de nouveaux blocs permanents au moment de la validation.
dirtyblocks	bigint	Nombre de blocs qui devaient être écrits dans le cadre de cette validation.
headers	bigint	Nombre d'en-têtes de bloc qui devaient être écrits dans le cadre de cette validation.
numxids	integer	Nombre de transactions DML actives.
oldestxid	bigint	XID de la transaction DML active la plus ancienne.

Nom de la colonne	Type de données	Description
extwritelatency	bigint	Information à utilisation interne uniquement.
metadatawritten	int	Information à utilisation interne uniquement.
tombstonedblocks	bigint	Information à utilisation interne uniquement.
tossedblocks	bigint	Information à utilisation interne uniquement.
batched_by	bigint	Information à utilisation interne uniquement.

Exemple de requête

```
select node, datediff(ms,startqueue,startwork) as queue_time,
datediff(ms, startwork, endtime) as commit_time, queuelen
from stl_commit_stats
where xid = 2574
order by node;

node | queue_time   | commit_time | queuelen
-----+--------------+-------------+---------
  -1 |            0 |         617 |        0
   0 | 444950725641 |         616 |        0
   1 | 444950725636 |         616 |        0
```

STL_CONNECTION_LOG

Enregistre les tentatives d'authentification, ainsi que les connexions et déconnexions.

STL_CONNECTION_LOG n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_CONNECTION_LOG. Les données de la vue de surveillance SYS sont formatées pour

être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
event	character(50)	Connexion ou événement d'authentification.
recordtime	timestamp	Heure de l'événement.
remotehost	character(45)	Nom ou adresse IP de l'hôte distant.
remoteport	character(32)	Numéro de port de l'hôte distant.
pid	entier	ID de processus associé à l'instruction.
dbname	character(50)	Nom de la base de données.
nom d'utilisateur	character(50)	Nom d'utilisateur.
authmethod	character(32)	Méthode d'authentification.
duration	entier	Durée de connexion en microsecondes.
sslversion	character(50)	Version du protocole SSL (Secure Sockets Layer). Cette colonne est vide lorsque le protocole SSL est désactivé.
sslcipher	character(128)	Chiffrement SSL.
mtu	entier	Unité de transmission maximale (MTU).
sslcompression	character(64)	Type de compression SSL.
sslexpansion	character(64)	Type d'extension SSL.
iamauthguid	character(36)	ID d'authentification IAM pour la CloudTrail demande.

Nom de la colonne	Type de données	Description
application_name	character(250)	Nom initial ou mis à jour de l'application pour une séance.
os_version	character(64)	La version du système d'exploitation de la machine cliente qui se connecte à votre cluster Amazon Redshift.
driver_version	character(64)	La version du pilote ODBC ou JDBC qui se connecte à votre cluster Amazon Redshift à partir de vos outils clients SQL tiers.
plugin_name	character(32)	Le nom du plugin utilisé pour se connecter à votre cluster Amazon Redshift.
protocol_version	entier	La version du protocole interne que le pilote Amazon Redshift utilise pour établir sa connexion avec le serveur. Les versions du protocole sont négociées entre le pilote et le serveur. La version décrit les fonctions disponibles. Les valeurs valides sont les suivantes : • 0 (BASE_SERVER_PROTOCOL_VERSION) • 1 (EXTENDED_RESULT_METADATA_SERVER_PROTOCOL_VERSION) – Pour enregistrer un aller-retour par requête, le serveur envoie des informations supplémentaires sur les métadonnées du jeu de résultats. • 2 (BINARY_PROTOCOL_VERSION) – Selon le type de données du jeu de résultats, le serveur envoie des données au format binaire. • 3 (EXTENDED2_RESULT_METADATA_SERVER_PROTOCOL_VERSION) — Le serveur envoie les informations de distinction majuscules/minuscules (classement) d'une colonne.

Nom de la colonne	Type de données	Description
sessionid	character(36)	Identifiant unique au niveau mondial pour la séance en cours. L'ID de séance persiste lors des redémarrages échoués du nœud.
compression	character(16)	Algorithme de compression utilisé pour la connexion.

Exemples de requêtes

Pour afficher les détails pour les connexions ouvertes, exécutez la requête suivante.

```
select recordtime, username, dbname, remotehost, remoteport
from stl_connection_log
where event = 'initiating session'
and pid not in
(select pid from stl_connection_log
where event = 'disconnecting session')
order by 1 desc;

recordtime          | username    | dbname     | remotehost    | remoteport

--------------------+-------------+------------+---------------+------------
2014-11-06 20:30:06 | rdsdb       | dev        | [local]       |

2014-11-06 20:29:37 | test001     | test       | 10.49.42.138  | 11111

2014-11-05 20:30:29 | rdsdb       | dev        | 10.49.42.138  | 33333

2014-11-05 20:28:35 | rdsdb       | dev        | [local]       |
(4 rows)
```

L'exemple suivant reflète un échec de tentative d'authentification, et une connexion et une déconnexion réussie.

```
select event, recordtime, remotehost, username
from stl_connection_log order by recordtime;
```

```
            event      |         recordtime        |  remotehost  | username

-----------------------+---------------------------+--------------+---------
authentication failure | 2012-10-25 14:41:56.96391 | 10.49.42.138 | john

authenticated          | 2012-10-25 14:42:10.87613 | 10.49.42.138 | john

initiating session     | 2012-10-25 14:42:10.87638 | 10.49.42.138 | john

disconnecting session  | 2012-10-25 14:42:19.95992 | 10.49.42.138 | john

(4 rows)
```

L'exemple suivant montre la version du pilote ODBC, le système d'exploitation de la machine cliente et le plugin utilisé pour se connecter au cluster Amazon Redshift. Dans cet exemple, le plugin utilisé est destiné à l'authentification du pilote ODBC standard à l'aide d'un nom de connexion et d'un mot de passe.

```
select driver_version, os_version, plugin_name from stl_connection_log;

driver_version                          |  os_version                       |
 plugin_name
-----------------------------------------+-----------------------------------
+--------------------
Amazon Redshift ODBC Driver 1.4.15.0001 | Darwin 18.7.0 x86_64              | none
Amazon Redshift ODBC Driver 1.4.15.0001 | Linux 4.15.0-101-generic x86_64   | none
```

L'exemple suivant montre la version du système d'exploitation de la machine cliente, la version du pilote et la version du protocole.

```
select os_version, driver_version, protocol_version from stl_connection_log;

os_version                      |  driver_version              | protocol_version
--------------------------------+------------------------------+--------------------
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
Linux 4.15.0-101-generic x86_64 | Redshift JDBC Driver 2.0.0.0 | 2
```

STL_DDLTEXT

Capture les instructions DDL suivantes qui ont été exécutées sur le système.

Ces instructions DDL incluent les requêtes et les objets suivants :

- CREATE SCHEMA, TABLE, VIEW
- DROP SCHEMA, TABLE, VIEW
- ALTER SCHEMA, TABLE

Consultez aussi STL_QUERYTEXT, STL_UTILITYTEXT et SVL_STATEMENTTEXT. Ces vues fournissent une chronologie des commandes SQL qui sont exécutées sur le système ; cet historique est utile pour le dépannage et la création d'un journal d'activité d'audit de toutes les activités du système.

Utilisez les colonnes STARTTIME et ENDTIME pour découvrir quelles instructions ont été enregistrées pendant une période donnée. Les longs blocs de texte SQL sont divisés en lignes de 200 caractères de long ; la colonne SEQUENCE identifie les fragments de texte qui appartiennent à une seule instruction.

STL_DDLTEXT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
xid	bigint	ID de transaction associé à l'instruction.
pid	entier	ID de processus associé à l'instruction.
étiquette	caractère (320)	Le nom du fichier utilisé pour exécuter la requête ou une étiquette définie avec une commande SET QUERY_GRO UP. Si la requête n'est pas basée sur un fichier ou si le

Nom de la colonne	Type de données	Description
		paramètre QUERY_GROUP n'est pas défini, le champ est vide.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **2009-06-12 11:29:19.131358** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **2009-06-12 11:29:19.131358** .
sequence	entier	Lorsqu'une seule instruction contient plus de 200 caractères, des lignes supplémentaires sont enregistrées pour l'instruction. Sequence 0 correspond à la première ligne, 1 à la deuxième, et ainsi de suite.
text	character(200)	Texte SQL, par incréments de 200 caractères. Ce champ peut contenir des caractères spéciaux tels qu'une barre oblique inverse (\\) et un caractère de saut de ligne (\n).

Exemples de requêtes

La requête suivante renvoie des enregistrements qui incluent des instructions DDL précédemment exécutées.

```
select xid, starttime, sequence, substring(text,1,40) as text
from stl_ddltext order by xid desc, sequence;
```

Voici un exemple de sortie montrant quatre instructions CREATE TABLE. Les instructions DDL apparaissent dans la colonne `text`, tronquée à des fins de lisibilité.

```
 xid  |         starttime          | sequence |                 text
------+----------------------------+----------
+----------------------------------------
```

```
 1806 | 2013-10-23 00:11:14.709851 |        0 | CREATE TABLE supplier ( s_suppkey int4
 N
 1806 | 2013-10-23 00:11:14.709851 |        1 |  s_comment varchar(101) NOT NULL )
 1805 | 2013-10-23 00:11:14.496153 |        0 | CREATE TABLE region ( r_regionkey int4
 N
 1804 | 2013-10-23 00:11:14.285986 |        0 | CREATE TABLE partsupp ( ps_partkey int8
 1803 | 2013-10-23 00:11:14.056901 |        0 | CREATE TABLE part ( p_partkey int8 NOT
 N
 1803 | 2013-10-23 00:11:14.056901 |        1 | ner char(10) NOT NULL , p_retailprice
 nu
(6 rows)
```

Reconstruction de SQL stocké

L'instruction SQL suivante répertorie les lignes stockées dans la colonne `text` STL_DDLTEXT. Les lignes sont classées par `xid` et `sequence`. Si le code SQL d'origine était plus long que 200 caractères, STL_DDLTEXT peut contenir plusieurs lignes par `sequence`.

```
SELECT xid, sequence, LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text)
 END, '') WITHIN GROUP (ORDER BY sequence) as query_statement
FROM stl_ddltext GROUP BY xid, sequence ORDER BY xid, sequence;
```

```
xid     |  sequence | query_statement
--------+-----------+-----------------
7886671    0          create external schema schema_spectrum_uddh\nfrom data catalog
\ndatabase 'spectrum_db_uddh'\niam_role ''\ncreate external database if not exists;
7886752    0          CREATE EXTERNAL TABLE schema_spectrum_uddh.soccer_league\n(\n
  league_rank smallint,\n  prev_rank   smallint,\n  club_name   varchar(15),\n
 league_name varchar(20),\n  league_off  decimal(6,2),\n  le
7886752    1          ague_def  decimal(6,2),\n  league_spi  decimal(6,2),\n
 league_nspi smallint\n)\nROW FORMAT DELIMITED \n    FIELDS TERMINATED BY ',' \n
 LINES TERMINATED BY '\\n\\l'\nstored as textfile\nLOCATION 's
7886752    2          3://mybucket-spectrum-uddh/'\ntable properties
 ('skip.header.line.count'='1');
...
```

Pour reconstruire le code SQL stocké dans la colonne `text` de STL_DDLTEXT, exécutez l'instruction SQL suivante. Il réunit les instructions DDL d'un ou de plusieurs segments dans la colonne `text`. Avant d'exécuter le SQL reconstruit, remplacez tout caractère spécial (\n) par un saut de ligne dans

votre client SQL. Les résultats de l'instruction SELECT suivante rassemblent trois lignes dans l'ordre de la séquence pour reconstituer le code SQL dans le champ `query_statement`.

```
SELECT LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END) WITHIN
 GROUP (ORDER BY sequence) as query_statement
FROM stl_ddltext GROUP BY xid, endtime order by xid, endtime;
```

```
query_statement
--------------
create external schema schema_spectrum_uddh\nfrom data catalog\ndatabase
 'spectrum_db_uddh'\niam_role ''\ncreate external database if not exists;
CREATE EXTERNAL TABLE schema_spectrum_uddh.soccer_league\n(\n  league_rank smallint,
\n  prev_rank   smallint,\n  club_name   varchar(15),\n  league_name varchar(20),\n
 league_off  decimal(6,2),\n  league_def  decimal(6,2),\n  league_spi  decimal(6,2),
\n  league_nspi smallint\n)\nROW FORMAT DELIMITED \n    FIELDS TERMINATED BY ',' \n
  LINES TERMINATED BY '\\n\\l'\nstored as textfile\nLOCATION 's3://mybucket-spectrum-
uddh/'\ntable properties ('skip.header.line.count'='1');
```

STL_DELETE

Analyse les étapes d'exécution de la suppression pour les requêtes.

STL_DELETE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_DELETE contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
tbl	entier	ID de table.

Exemples de requêtes

Afin de créer une ligne dans STL_DELETE, l'exemple suivant insère une ligne dans la table EVENT, puis la supprime.

Tout d'abord, insérez une ligne dans la table EVENT et vérifiez qu'elle a été insérée.

```
insert into event(eventid,venueid,catid,dateid,eventname)
values ((select max(eventid)+1 from event),95,9,1857,'Lollapalooza');
```

```
select * from event
where eventname='Lollapalooza'
order by eventid;
```

```
 eventid | venueid | catid | dateid |  eventname   |      starttime
---------+---------+-------+--------+--------------+---------------------
    4274 |     102 |     9 |   1965 | Lollapalooza | 2008-05-01 19:00:00
    4684 |     114 |     9 |   2105 | Lollapalooza | 2008-10-06 14:00:00
    5673 |     128 |     9 |   1973 | Lollapalooza | 2008-05-01 15:00:00
    5740 |      51 |     9 |   1933 | Lollapalooza | 2008-04-17 15:00:00
    5856 |     119 |     9 |   1831 | Lollapalooza | 2008-01-05 14:00:00
    6040 |     126 |     9 |   2145 | Lollapalooza | 2008-11-15 15:00:00
    7972 |      92 |     9 |   2026 | Lollapalooza | 2008-07-19 19:30:00
    8046 |      65 |     9 |   1840 | Lollapalooza | 2008-01-14 15:00:00
    8518 |      48 |     9 |   1904 | Lollapalooza | 2008-03-19 15:00:00
    8799 |      95 |     9 |   1857 | Lollapalooza |
(10 rows)
```

Maintenant, supprimez la ligne que vous avez ajoutée à la table EVENT et vérifiez qu'elle a été supprimée.

```
delete from event
where eventname='Lollapalooza' and eventid=(select max(eventid) from event);
```

```
select * from event
where eventname='Lollapalooza'
order by eventid;
```

```
 eventid | venueid | catid | dateid |  eventname   |      starttime
---------+---------+-------+--------+--------------+---------------------
    4274 |     102 |     9 |   1965 | Lollapalooza | 2008-05-01 19:00:00
    4684 |     114 |     9 |   2105 | Lollapalooza | 2008-10-06 14:00:00
    5673 |     128 |     9 |   1973 | Lollapalooza | 2008-05-01 15:00:00
    5740 |      51 |     9 |   1933 | Lollapalooza | 2008-04-17 15:00:00
```

```
    5856 |     119 |     9 |   1831 | Lollapalooza | 2008-01-05 14:00:00
    6040 |     126 |     9 |   2145 | Lollapalooza | 2008-11-15 15:00:00
    7972 |      92 |     9 |   2026 | Lollapalooza | 2008-07-19 19:30:00
    8046 |      65 |     9 |   1840 | Lollapalooza | 2008-01-14 15:00:00
    8518 |      48 |     9 |   1904 | Lollapalooza | 2008-03-19 15:00:00
(9 rows)
```

Puis interrogez stl_delete pour voir les étapes d'exécution de la suppression. Dans cet exemple, comme la requête a retourné plus de 300 lignes, la sortie ci-dessous est raccourcie à des fins d'affichage.

```
select query, slice, segment, step, tasknum, rows, tbl from stl_delete order by query;
```

```
 query | slice | segment | step | tasknum | rows |  tbl
-------+-------+---------+------+---------+------+--------
     7 |     0 |       0 |    1 |       0 |    0 | 100000
     7 |     1 |       0 |    1 |       0 |    0 | 100000
     8 |     0 |       0 |    1 |       2 |    0 | 100001
     8 |     1 |       0 |    1 |       2 |    0 | 100001
     9 |     0 |       0 |    1 |       4 |    0 | 100002
     9 |     1 |       0 |    1 |       4 |    0 | 100002
    10 |     0 |       0 |    1 |       6 |    0 | 100003
    10 |     1 |       0 |    1 |       6 |    0 | 100003
    11 |     0 |       0 |    1 |       8 |    0 | 100253
    11 |     1 |       0 |    1 |       8 |    0 | 100253
    12 |     0 |       0 |    1 |       0 |    0 | 100255
    12 |     1 |       0 |    1 |       0 |    0 | 100255
    13 |     0 |       0 |    1 |       2 |    0 | 100257
    13 |     1 |       0 |    1 |       2 |    0 | 100257
    14 |     0 |       0 |    1 |       4 |    0 | 100259
    14 |     1 |       0 |    1 |       4 |    0 | 100259
    ...
```

STL_DISK_FULL_DIAG

Informations de journaux sur les erreurs enregistrées lorsque le disque est plein.

STL_DISK_FULL_DIAG n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
currenttime	bigint	Jour et heure de génération de l'erreur, en microsecondes depuis le 1er janvier 2000.
node_num	bigint	Identifiant du nœud.
query_id	bigint	Identifiant de la requête responsable de l'erreur.
temp_blocks	bigint	Nombre de blocs temporaires créés par la requête.

Exemples de requêtes

L'exemple suivant renvoie des détails sur les données stockées en cas d'erreur de disque plein.

```
select * from stl_disk_full_diag
```

L'exemple suivant convertit `currenttime` en horodatage.

```
select '2000-01-01'::timestamp + (currenttime/1000000.0)* interval '1 second' as
 currenttime,node_num,query_id,temp_blocks from pg_catalog.stl_disk_full_diag;
```

```
        currenttime         | node_num | query_id | temp_blocks
----------------------------+----------+----------+-------------
 2019-05-18 19:19:18.609338 |        0 |   569399 |       70982
 2019-05-18 19:37:44.755548 |        0 |   569580 |       70982
 2019-05-20 13:37:20.566916 |        0 |   597424 |       70869
```

STL_DIST

Enregistre les informations sur l'activité réseau pendant l'exécution des étapes de requête qui distribuent les données. Le trafic réseau est capturé par le nombre de lignes, d'octets et de paquets envoyés sur le réseau pendant une étape donnée sur une tranche donnée. La durée de l'étape est la différence entre les heures de début et de fin enregistrées.

Pour identifier les étapes de distribution d'une requête, recherchez les étiquettes dist de la vue QUERY_SUMMARY, ou exécutez la commande EXPLAIN et recherchez les attributs de l'étape qui incluent dist.

STL_DIST est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_DIST contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.

Nom de la colonne	Type de données	Description
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **2009-06-12 11:29:19.131358** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **2009-06-12 11:29:19.131358** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
octets	bigint	Taille, en octets, de toutes les lignes de sortie de l'étape.
paquets	entier	Nombre total de paquets envoyés sur le réseau.

Exemples de requêtes

L'exemple suivant renvoie les informations de distribution pour les requêtes avec un ou plusieurs paquets, et une durée supérieure à zéro.

```
select query, slice, step, rows, bytes, packets,
datediff(seconds, starttime, endtime) as duration
from stl_dist
where packets>0 and datediff(seconds, starttime, endtime)>0
order by query
limit 10;
```

```
 query  | slice | step |  rows  |  bytes   | packets | duration
--------+-------+------+--------+---------+---------+----------
    567 |     1 |    4 |  49990 | 6249564 |     707 |        1
    630 |     0 |    5 |   8798 |  408404 |      46 |        2
    645 |     1 |    4 |   8798 |  408404 |      46 |        1
    651 |     1 |    5 | 192497 | 9226320 |    1039 |        6
    669 |     1 |    4 | 192497 | 9226320 |    1039 |        4
    675 |     1 |    5 |   3766 |  194656 |      22 |        1
    696 |     0 |    4 |   3766 |  194656 |      22 |        1
    705 |     0 |    4 |    930 |   44400 |       5 |        1
 111525 |     0 |    3 |     68 |   17408 |       2 |        1
(9 rows)
```

STL_ERROR

Enregistre les erreurs de traitement interne générées par le moteur de base de données Amazon Redshift. STL_ERROR n'enregistre pas les erreurs ou les messages SQL. Les informations contenues dans STL_ERROR sont utiles pour le dépannage de certaines erreurs. Un ingénieur de AWS support peut vous demander de fournir ces informations dans le cadre du processus de dépannage.

STL_ERROR est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Pour obtenir la liste des codes d'erreur qui peuvent être générés lors du chargement des données avec la commande Copy, consultez Référence des erreurs de chargement.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.

Nom de la colonne	Type de données	Description
process	character(12)	Processus qui a levé l'exception.
recordtime	timestamp	Heure à laquelle l'erreur s'est produite.
pid	entier	ID du processus. Le STL_QUERY tableau contient le processus IDs et la requête unique IDs pour les requêtes terminées.
errcode	entier	Code d'erreur correspondant à la catégorie d'erreur.
dans le fichier	character(90)	Nom du fichier source où l'erreur s'est produite.
linenum	entier	Numéro de ligne du fichier source où l'erreur s'est produite.
context	character(100)	Cause de l'erreur.
error	character(512)	Message d'erreur.

Exemples de requêtes

L'exemple suivant récupère les informations d'erreur dans STL_ERROR.

```
select process, errcode, linenum as line,
trim(error) as err
from stl_error;

   process    | errcode | line |                                 err
--------------+---------+------
+----------------------------------------------------------------------
 padbmaster   |    8001 |  194 | Path prefix: s3://redshift-downloads/testnulls/
venue.txt*
 padbmaster   |    8001 |  529 | Listing bucket=redshift-downloads prefix=tests/
category-csv-quotes
 padbmaster   |       2 |  190 | database "template0" is not currently accepting
 connections
 padbmaster   |      32 | 1956 | pq_flush: could not send data to client: Broken pipe
```

```
(4 rows)
```

STL_EXPLAIN

Affiche le plan EXPLAIN pour une requête qui a été soumise à exécution.

STL_EXPLAIN est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_EXPLAIN contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_EXPLAIN Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
nodeid	entier	Identificateur de nœud de plan, où un nœud correspond à une ou plusieurs étapes de l'exécution de la requête.
parentid	entier	Identificateur de nœud de plan d'un nœud parent. Un nœud parent a un certain nombre de nœuds enfants. Par exemple, une jointure de fusion est le parent des analyses des tables jointes.

Nom de la colonne	Type de données	Description
plannode	character(400)	Texte du nœud de la sortie EXPLAIN. Les nœuds de plan qui font référence à l'exécution sur les nœuds de calcul sont préfixés par **XN** dans la sortie EXPLAIN.
info	character(400)	Informations de qualificateur et de filtre pour le nœud de plan. Par exemple, les conditions de jointure et les restrictions clause WHERE sont incluses dans cette colonne.

Exemples de requêtes

Considérons la sortie EXPLAIN suivante d'une requête de jointure d'agrégation :

```
explain select avg(datediff(day, listtime, saletime)) as avgwait
from sales, listing where sales.listid = listing.listid;
                                QUERY PLAN

------------------------------------------------------------------------------
 XN Aggregate  (cost=6350.30..6350.31 rows=1 width=16)
  ->  XN Hash Join DS_DIST_NONE  (cost=47.08..6340.89 rows=3766 width=16)
        Hash Cond: ("outer".listid = "inner".listid)
        -> XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=12)
        -> XN Hash  (cost=37.66..37.66 rows=3766 width=12)
              -> XN Seq Scan on sales  (cost=0.00..37.66 rows=3766 width=12)
(6 rows)
```

Si vous exécutez cette requête et que son ID de requête est 10, vous pouvez utiliser la table STL_EXPLAIN pour afficher le même type d'informations que renvoie la commande EXPLAIN :

```
select query,nodeid,parentid,substring(plannode from 1 for 30),
substring(info from 1 for 20) from stl_explain
where query=10 order by 1,2;

query| nodeid |parentid|           substring            |    substring
-----+--------+--------+--------------------------------+------------------
10   |      1 |      0 |XN Aggregate  (cost=6717.61..6  |
10   |      2 |      1 |  -> XN Merge Join DS_DIST_NO   | Merge Cond:("outer"
10   |      3 |      2 |       -> XN Seq Scan on lis    |
```

```
10   |      4 |      2 |       -> XN Seq Scan on sal    |
(4 rows)
```

Considérons la requête suivante :

```
select event.eventid, sum(pricepaid)
from event, sales
where event.eventid=sales.eventid
group by event.eventid order by 2 desc;

eventid |   sum
--------+----------
    289 | 51846.00
   7895 | 51049.00
   1602 | 50301.00
    851 | 49956.00
   7315 | 49823.00
...
```

Si l'ID de requête est le 15, la requête de la vue système suivante renvoie les nœuds de plan qui ont été exécutés. Dans ce cas, l'ordre des nœuds est inversé pour afficher l'ordre réel d'exécution :

```
select query,nodeid,parentid,substring(plannode from 1 for 56)
from stl_explain where query=15 order by 1, 2 desc;

query|nodeid|parentid|                          substring
-----+------+--------+-----------------------------------------------------------
15   |    8 |      7 |                                -> XN Seq Scan on eve
15   |    7 |      5 |                          -> XN Hash(cost=87.98..87.9
15   |    6 |      5 |                          -> XN Seq Scan on sales(cos
15   |    5 |      4 |                    -> XN Hash Join DS_DIST_OUTER(cos
15   |    4 |      3 |              -> XN HashAggregate(cost=862286577.07..
15   |    3 |      2 |        -> XN Sort(cost=1000862287175.47..10008622871
15   |    2 |      1 |  -> XN Network(cost=1000862287175.47..1000862287197.
15   |    1 |      0 |XN Merge(cost=1000862287175.47..1000862287197.46 rows=87
(8 rows)
```

La requête suivante extrait la requête IDs pour tous les plans de requête contenant une fonction de fenêtre :

```
select query, trim(plannode) from stl_explain
where plannode like '%Window%';
```

```
query|                                    btrim
-----+-------------------------------------------------------------------------
26   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
27   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
(2 rows)
```

STL_FILE_SCAN

Renvoie les fichiers qu'Amazon Redshift lit pendant le chargement des données à l'aide de la commande COPY.

L'interrogation de cette vue peut aider à résoudre les erreurs de chargement de données. STL_FILE_SCAN peut être particulièrement utile pour identifier les problèmes de charges parallèles de données, car celles-ci chargent généralement de nombreux fichiers avec une seule commande COPY.

STL_FILE_SCAN est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_FILE_SCAN contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_LOAD_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.

Nom de la colonne	Type de données	Description
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
name	character(90)	Chemin d'accès complet et nom du fichier qui a été chargé.
lines	bigint	Nombre de lignes lues dans le fichier.
octets	bigint	Nombre d'octets lus dans le fichier.
loadtime	bigint	Durée de chargement du fichier (en microsecondes).
curtime	Horodatage	Horodatage correspondant à l'heure à laquelle Amazon Redshift a commencé à traiter le fichier.
is_partial	entier	Si true = 1, cette valeur indique que le fichier d'entrée est divisé en plages lors d'une opération COPY. Si false = 0, le fichier d'entrée n'est pas divisé.
start_offset	bigint	Si le fichier d'entrée est fractionné lors d'une opération COPY, cela indique la valeur de décalage du fractionnement (en octets). Si le fichier n'est pas fractionné, cette valeur est réglée sur 0.

Exemples de requêtes

La requête suivante extrait les noms et les durées de chargement de tous les fichiers qui ont nécessité plus de 1 000 000 microsecondes pour être lus par Amazon Redshift.

```
select trim(name)as name, loadtime from stl_file_scan
where loadtime > 1000000;
```

Cette requête renvoie l'exemple de sortie suivant :

```
           name             | loadtime
----------------------------+----------
 listings_pipe.txt          |  9458354
 allusers_pipe.txt          |  2963761
 allevents_pipe.txt         |  1409135
 tickit/listings_pipe.txt   |  7071087
 tickit/allevents_pipe.txt  |  1237364
 tickit/allusers_pipe.txt   |  2535138
 listings_pipe.txt          |  6706370
 allusers_pipe.txt          |  3579461
 allevents_pipe.txt         |  1313195
 tickit/allusers_pipe.txt   |  3236060
 tickit/listings_pipe.txt   |  4980108
(11 rows)
```

STL_HASH

Analyse les étapes d'exécution du hachage pour les requêtes.

STL_HASH est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_HASH contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.

Nom de la colonne	Type de données	Description
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
octets	bigint	Taille, en octets, de toutes les lignes de sortie de l'étape.
slots	entier	Nombre total de compartiments de hachage.
occupied	entier	Nombre total d'emplacements qui contiennent des enregistrements.
maxlength	entier	Taille de l'emplacement le plus grand.
tbl	entier	ID de table.

Nom de la colonne	Type de données	Description
is_diskbased	character(1)	Si la valeur est true (t), la requête a été exécutée en tant qu'opération sur disque. Si elle est false (f), la requête a été exécutée en mémoire.
workmem	bigint	Nombre total d'octets de mémoire de travail assignés à l'étape.
num_parts	entier	Nombre total de partitions dans lesquelles une table de hachage a été divisée pendant une étape de hachage.
est_rows	bigint	Estimation du nombre de lignes à hacher.
num_blocks_permitted	entier	Information à utilisation interne uniquement.
redimensionne	entier	Information à utilisation interne uniquement.
checksum	bigint	Information à utilisation interne uniquement.
runtime_filter_size	entier	La taille du filtre d'exécution en octets.
max_runtime_filter_size	entier	La taille maximale du filtre d'exécution en octets.

Exemples de requêtes

L'exemple suivant renvoie des informations sur le nombre de partitions utilisées dans un hachage pour la requête 720 et indique qu'aucune des étapes ne s'est exécutée sur le disque.

```
select slice, rows, bytes, occupied, workmem, num_parts, est_rows,
 num_blocks_permitted, is_diskbased
from stl_hash
```

```
where query=720 and segment=5
order by slice;
```

```
 slice | rows | bytes  | occupied | workmem  | num_parts | est_rows |
 num_blocks_permitted | is_diskbased
-------+------+--------+----------+----------+-----------+----------
+----------------------+--------------
     0 |  145 | 585800 |        1 | 88866816 |        16 |        1 |
 52              f
     1 |    0 |      0 |        0 |        0 |        16 |        1 |
 52              f
(2 rows)
```

STL_HASHJOIN

Analyse les étapes d'exécution de la jointure de hachage pour les requêtes.

STL_HASHJOIN est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_HASHJOIN contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.

Nom de la colonne	Type de données	Description
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
tbl	entier	ID de table.
num_parts	entier	Nombre total de partitions dans lesquelles une table de hachage a été divisée pendant une étape de hachage.

Nom de la colonne	Type de données	Description
join_type	entier	Type de jointure pour l'étape : • 0. La requête a utilisé une jointure interne. • 1. La requête a utilisé une jointure externe gauche. • 2. La requête a utilisé une jointure externe complète. • 3. La requête a utilisé une jointure externe droite. • 4. La requête a utilisé un opérateur UNION. • 5. La requête a utilisé une condition IN. • 6. Information à utilisation interne uniquement. • 7. Information à utilisation interne uniquement. • 8. Information à utilisation interne uniquement. • 9. Information à utilisation interne uniquement. • 10. Information à utilisation interne uniquement. • 11. Information à utilisation interne uniquement. • 12. Information à utilisation interne uniquement.
hash_looped	character(1)	Information à utilisation interne uniquement.
switched_parts	character(1)	Indique s'il y a eu permutation des côtés externe et interne.
used_prefetching	character(1)	Information à utilisation interne uniquement.
hash_segment	entier	Segment de l'étape de hachage correspondante.
hash_step	entier	Numéro d'étape de l'étape de hachage correspondante.
checksum	bigint	Information à utilisation interne uniquement.
distribution	entier	Information à utilisation interne uniquement.

Exemples de requêtes

L'exemple suivant renvoie le nombre de partitions utilisées dans une jointure de hachage pour la requête 720.

```
select query, slice, tbl, num_parts
from stl_hashjoin
where query=720 limit 10;
```

```
 query | slice | tbl | num_parts
-------+-------+-----+-----------
   720 |     0 | 243 |         1
   720 |     1 | 243 |         1
(2 rows)
```

STL_INSERT

Analyse les étapes d'exécution de l'insertion pour les requêtes.

STL_INSERT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_INSERT contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
tbl	entier	ID de table.
inserted_mega_value	character(1)	Information à utilisation interne uniquement. Ces informations indiquent si l'étape d'insertion donnée a inséré une valeur élevée. Une valeur élevée sera stockée dans plusieurs blocs. La taille de bloc par défaut étant d'1 Mo,

Nom de la colonne	Type de données	Description
		une valeur importante est supérieure à 1 Mo dans une configuration par défaut.

Exemples de requêtes

L'exemple suivant renvoie les étapes d'exécution de l'insertion pour la requête la plus récente.

```
select slice, segment, step, tasknum, rows, tbl
from stl_insert
where query=pg_last_query_id();
```

```
 slice | segment | step | tasknum | rows  |  tbl
-------+---------+------+---------+-------+--------
     0 |       2 |    2 |      15 | 24958 | 100548
     1 |       2 |    2 |      15 | 25032 | 100548
(2 rows)
```

STL_LIMIT

Analyse les étapes de l'exécution qui se produisent lorsqu'une clause LIMIT est utilisée dans une requête SELECT.

STL_LIMIT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_LIMIT contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
checksum	bigint	Information à utilisation interne uniquement.

Exemples de requêtes

Pour pouvoir générer une ligne dans STL_LIMIT, cet exemple exécute d'abord la requête suivante sur la table VENUE l'aide de la clause LIMIT.

```
select * from venue
order by 1
limit 10;
```

```
 venueid |         venuename          |    venuecity    | venuestate | venueseats
---------+----------------------------+-----------------+------------+------------
       1 | Toyota Park                | Bridgeview      | IL         |          0
       2 | Columbus Crew Stadium      | Columbus        | OH         |          0
       3 | RFK Stadium                | Washington      | DC         |          0
       4 | CommunityAmerica Ballpark  | Kansas City     | KS         |          0
       5 | Gillette Stadium           | Foxborough      | MA         |      68756
       6 | New York Giants Stadium    | East Rutherford | NJ         |      80242
       7 | BMO Field                  | Toronto         | ON         |          0
       8 | The Home Depot Center      | Carson          | CA         |          0
       9 | Dick's Sporting Goods Park | Commerce City   | CO         |          0
      10 | Pizza Hut Park             | Frisco          | TX         |          0
(10 rows)
```

Ensuite, exécutez la requête suivante pour rechercher l'ID de la dernière requête que vous avez exécutée sur la table VENUE.

```
select max(query)
from stl_query;
```

```
  max
--------
 127128
(1 row)
```

Le cas échéant, vous pouvez exécuter la requête suivante pour vérifier que l'ID de requête correspond à la requête LIMIT que vous avez exécutée précédemment.

```
select query, trim(querytxt)
from stl_query
where query=127128;
```

```
 query  |                 btrim
--------+----------------------------------------
 127128 | select * from venue order by 1 limit 10;
```

```
(1 row)
```

Enfin, exécutez la requête suivante pour renvoyer les informations sur la requête LIMIT à partir de la table STL_LIMIT.

```
select slice, segment, step, starttime, endtime, tasknum
from stl_limit
where query=127128
order by starttime, endtime;
```

```
  slice | segment | step |         starttime          |          endtime           |
 tasknum
 -------+---------+------+----------------------------+----------------------------
+---------
      1 |       1 |    3 | 2013-09-06 22:56:43.608114 | 2013-09-06 22:56:43.609383 |
   15
      0 |       1 |    3 | 2013-09-06 22:56:43.608708 | 2013-09-06 22:56:43.609521 |
   15
  10000 |       2 |    2 | 2013-09-06 22:56:43.612506 | 2013-09-06 22:56:43.612668 |
    0
(3 rows)
```

STL_LOAD_COMMITS

renvoie les informations pour suivre ou dépanner une charge de données.

Cette vue enregistre la progression de chaque fichier de données lorsqu'il est chargé dans une table de base de données.

STL_LOAD_COMMIS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_LOAD_COMMITS contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous

recommandons d'utiliser la vue de surveillance SYS. SYS_LOAD_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Tranche chargée pour cette entrée.
name	character(256)	Valeur définie par le système.
filename	character(256)	Nom du fichier suivi.
byte_offset	entier	Information à utilisation interne uniquement.
lines_scanned	entier	Nombre de lignes du fichier de charge analysées. Ce nombre peut ne pas correspondre au nombre de lignes qui sont effectivement chargées. Par exemple, la charge peut analyser, mais tolérer, un certain nombre d'enregistrements incorrects, en fonction de l'option MAXERROR de la commande COPY.
erreurs	entier	Information à utilisation interne uniquement.
curtime	timestamp	Heure à laquelle cette entrée a été mise à jour pour la dernière fois.
status	entier	Information à utilisation interne uniquement.
file_format	character(16)	Format du fichier de chargement. Les valeurs possibles sont les suivantes : • Avro

Nom de la colonne	Type de données	Description
		• JSON • ORC • Parquet • Texte
is_partial	entier	Si true = 1, cette valeur indique que le fichier d'entrée est divisé en plages lors d'une opération COPY. Si false = 0, le fichier d'entrée n'est pas divisé.
start_offset	bigint	Si le fichier d'entrée est fractionné lors d'une opération COPY, cela indique la valeur de décalage du fractionnement (en octets). Chaque fractionnement de fichier est journalisé en tant qu'enregistrement distinct avec la valeur start_offset correspondante. Si le fichier n'est pas fractionné, cette valeur est réglée sur 0.
copy_job_id	bigint	Identifiant de la tâche de copie. Un `0` indique qu'il n'y a aucun identifiant de tâche.

Exemples de requêtes

L'exemple suivant renvoie les détails de la dernière opération COPY.

```
select query, trim(filename) as file, curtime as updated
from stl_load_commits
where query = pg_last_copy_id();

 query |               file                |          updated
-------+-----------------------------------+----------------------------
 28554 | s3://dw-tickit/category_pipe.txt | 2013-11-01 17:14:52.648486
(1 row)
```

La requête suivante contient les entrées d'un nouveau chargement de tables dans la base de données TICKIT :

```
select query, trim(filename), curtime
```

```
from stl_load_commits
where filename like '%tickit%' order by query;
```

```
 query |           btrim           |          curtime
-------+---------------------------+---------------------------
 22475 | tickit/allusers_pipe.txt  | 2013-02-08 20:58:23.274186
 22478 | tickit/venue_pipe.txt     | 2013-02-08 20:58:25.070604
 22480 | tickit/category_pipe.txt  | 2013-02-08 20:58:27.333472
 22482 | tickit/date2008_pipe.txt  | 2013-02-08 20:58:28.608305
 22485 | tickit/allevents_pipe.txt | 2013-02-08 20:58:29.99489
 22487 | tickit/listings_pipe.txt  | 2013-02-08 20:58:37.632939
 22593 | tickit/allusers_pipe.txt  | 2013-02-08 21:04:08.400491
 22596 | tickit/venue_pipe.txt     | 2013-02-08 21:04:10.056055
 22598 | tickit/category_pipe.txt  | 2013-02-08 21:04:11.465049
 22600 | tickit/date2008_pipe.txt  | 2013-02-08 21:04:12.461502
 22603 | tickit/allevents_pipe.txt | 2013-02-08 21:04:14.785124
 22605 | tickit/listings_pipe.txt  | 2013-02-08 21:04:20.170594

(12 rows)
```

Le fait qu'un enregistrement soit écrit dans le fichier journal pour cette vue système ne signifie pas que la charge a été validée avec succès dans le cadre de la transaction contenante. Pour vérifier les validations de charge, interrogez la vue STL_UTILITYTEXT et recherchez l'enregistrement COMMIT qui correspond à une transaction COPY. Par exemple, cette requête joint STL_LOAD_COMMITS et STL_QUERY à partir d'une sous-requête sur STL_UTILITYTEXT :

```
select l.query,rtrim(l.filename),q.xid
from stl_load_commits l, stl_query q
where l.query=q.query
and exists
(select xid from stl_utilitytext where xid=q.xid and rtrim("text")='COMMIT');

 query |           rtrim           |  xid
-------+---------------------------+-------
 22600 | tickit/date2008_pipe.txt  | 68311
 22480 | tickit/category_pipe.txt  | 68066
  7508 | allusers_pipe.txt         | 23365
  7552 | category_pipe.txt         | 23415
  7576 | allevents_pipe.txt        | 23429
  7516 | venue_pipe.txt            | 23390
  7604 | listings_pipe.txt         | 23445
 22596 | tickit/venue_pipe.txt     | 68309
```

```
 22605 | tickit/listings_pipe.txt  | 68316
 22593 | tickit/allusers_pipe.txt  | 68305
 22485 | tickit/allevents_pipe.txt | 68071
  7561 | allevents_pipe.txt        | 23429
  7541 | category_pipe.txt         | 23415
  7558 | date2008_pipe.txt         | 23428
 22478 | tickit/venue_pipe.txt     | 68065
   526 | date2008_pipe.txt         |  2572
  7466 | allusers_pipe.txt         | 23365
 22482 | tickit/date2008_pipe.txt  | 68067
 22598 | tickit/category_pipe.txt  | 68310
 22603 | tickit/allevents_pipe.txt | 68315
 22475 | tickit/allusers_pipe.txt  | 68061
   547 | date2008_pipe.txt         |  2572
 22487 | tickit/listings_pipe.txt  | 68072
  7531 | venue_pipe.txt            | 23390
  7583 | listings_pipe.txt         | 23445
(25 rows)
```

Les exemples suivants mettent en évidence les valeurs de colonne is_partial et start_offset.

```
-- Single large file copy without scan range
SELECT count(*) FROM stl_load_commits WHERE query = pg_last_copy_id();
1

-- Single large uncompressed, delimited file copy with scan range
SELECT count(*) FROM stl_load_commits WHERE query = pg_last_copy_id();
16

-- Scan range offset logging in the file at 64MB boundary.
SELECT start_offset FROM stl_load_commits
WHERE query = pg_last_copy_id() ORDER BY start_offset;
0
67108864
134217728
201326592
268435456
335544320
402653184
469762048
536870912
603979776
671088640
```

```
738197504
805306368
872415232
939524096
1006632960
```

STL_LOAD_ERRORS

Affiche les enregistrements de toutes les erreurs de charge Amazon Redshift.

STL_LOAD_ERRORS contient un historique de toutes les erreurs de charge Amazon Redshift. Consultez Référence des erreurs de chargement pour une liste complète des erreurs de charge possibles et de leurs explications.

Interrogez STL_LOADERROR_DETAIL pour plus de détails, tels que la ligne et la colonne de données exactes où une erreur d'analyse s'est produite, une fois que vous avez interrogé STL_LOAD_ERRORS pour trouver des informations générales sur l'erreur.

STL_LOAD_ERRORS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_LOAD_ERRORS contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_LOAD_ERROR_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
slice	entier	Tranche où l'erreur s'est produite.
tbl	entier	ID de table.
starttime	timestamp	Heure de début de la charge au format UTC.
séance	entier	ID de séance pour la séance effectuant le chargement.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
filename	character(256)	Chemin d'accès complet du fichier d'entrée de la charge.
line_number	bigint	Numéro de ligne dans le fichier de chargement avec l'erreur. Pour l'opération COPY depuis JSON, le numéro de ligne de la dernière ligne de l'objet JSON avec l'erreur.
colname	character(127)	Champ avec l'erreur.
type	character(10)	Type de données du champ.
col_length	character(10)	Longueur de la colonne, le cas échéant. Ce champ est rempli lorsque le type de données a une longueur limite. Par exemple, pour une colonne avec le type de données « character(3) », cette colonne contient la valeur « 3 ».
position	entier	Position de l'erreur dans le champ.
raw_line	character(1024)	Données de chargement brutes qui contiennent l'erreur. Les caractères multioctets des données de chargement sont remplacés par un point.

Nom de la colonne	Type de données	Description
raw_field_value	char(1024)	Valeur de pré-analyse pour le champ « colname » qui a conduit à l'erreur d'analyse.
err_code	entier	Code d'erreur.
err_reason	character(100)	Explication de l'erreur.
is_partial	entier	Si true = 1, cette valeur indique que le fichier d'entrée est divisé en plages lors d'une opération COPY. Si false = 0, le fichier d'entrée n'est pas divisé.
start_offset	bigint	Si le fichier d'entrée est fractionné lors d'une opération COPY, cela indique la valeur de décalage du fractionnement (en octets). Si le numéro de ligne du fichier est inconnu, le numéro de ligne est -1. Si le fichier n'est pas fractionné, cette valeur est réglée sur 0.
copy_job_id	bigint	Identifiant de la tâche de copie. Un `0` indique qu'il n'y a aucun identifiant de tâche.

Exemples de requêtes

La requête suivante joint STL_LOAD_ERRORS et STL_LOADERROR_DETAIL pour afficher les erreurs détaillées qui se sont produites lors du chargement le plus récent.

```
select d.query, substring(d.filename,14,20),
d.line_number as line,
substring(d.value,1,16) as value,
substring(le.err_reason,1,48) as err_reason
from stl_loaderror_detail d, stl_load_errors le
where d.query = le.query
and d.query = pg_last_copy_id();

 query |    substring      | line |  value   |             err_reason
-------+-------------------+------+----------+----------------------------
    558| allusers_pipe.txt |  251 | 251      | String contains invalid or
                                              unsupported UTF8 code
```

```
    558| allusers_pipe.txt |  251 | ZRU29FGR | String contains invalid or
                                              unsupported UTF8 code
    558| allusers_pipe.txt |  251 | Kaitlin  | String contains invalid or
                                              unsupported UTF8 code
    558| allusers_pipe.txt |  251 | Walter   | String contains invalid or
                                              unsupported UTF8 code
```

L'exemple suivant utilise STL_LOAD_ERRORS avec STV_TBL_PERM pour créer une vue et utilise ensuite cette vue pour déterminer quelles erreurs se sont produites pendant le chargement des données dans la table EVENT :

```
create view loadview as
(select distinct tbl, trim(name) as table_name, query, starttime,
trim(filename) as input, line_number, colname, err_code,
trim(err_reason) as reason
from stl_load_errors sl, stv_tbl_perm sp
where sl.tbl = sp.id);
```

Ensuite, la requête suivante renvoie la dernière erreur qui s'est produite pendant le chargement de la table EVENT :

```
select table_name, query, line_number, colname, starttime,
trim(reason) as error
from loadview
where table_name ='event'
order by line_number limit 1;
```

La requête renvoie la dernière erreur de chargement qui s'est produite pour la table EVENT. Si aucune erreur de chargement ne s'est produite, la requête renvoie zéro ligne. Dans cet exemple, la requête renvoie une seule erreur :

```
 table_name | query | line_number | colname | error | starttime
------+-----+----+----+-------------------------------------------------------
+----------------------
event | 309 |  0 |  5 | Error in Timestamp value or format [%Y-%m-%d %H:%M:%S] |
 2014-04-22 15:12:44

(1 row)
```

Dans les cas où la commande COPY divise automatiquement des données de fichier volumineuses, non compressées et délimitées par du texte pour faciliter le parallélisme,les colonnes line_number,

is_partial et start_offset affichent des informations relatives aux fractionnements. (Le numéro de ligne peut être inconnu dans le cas où le numéro de ligne du fichier d'origine n'est pas disponible.)

```
--scan ranges information
SELECT line_number, POSITION, btrim(raw_line), btrim(raw_field_value),
btrim(err_reason), is_partial, start_offset FROM stl_load_errors
WHERE query = pg_last_copy_id();

--result
-1,51,"1008771|13463413|463414|2|28.00|38520.72|0.06|0.07|NO|1998-08-30|1998-09-25|
1998-09-04|TAKE BACK RETURN|RAIL|ans cajole sly","NO","Char length exceeds DDL
 length",1,67108864
```

STL_LOADERROR_DETAIL

Affiche le journal des erreurs d'analyse des données qui se sont produites lors de l'utilisation d'une commande COPY pour charger les tables. Afin d'économiser de l'espace sur le disque, 20 erreurs au plus par tranche de nœud sont enregistrées pour chaque opération de chargement.

Une erreur d'analyse se produit lorsqu'Amazon Redshift ne peut pas analyser un champ d'une ligne de données lors de son chargement dans une table. Par exemple, si une colonne de table attend un type de données entier et que le fichier de données contient une chaîne de lettres dans ce champ, il s'ensuit une erreur d'analyse.

Interrogez STL_LOADERROR_DETAIL pour plus de détails, tels que la ligne et la colonne de données exactes où une erreur d'analyse s'est produite, une fois que vous avez interrogé STL_LOAD_ERRORS pour trouver des informations générales sur l'erreur.

La vue STL_LOADERROR_DETAIL contient toutes les colonnes de données jusqu'à la colonne incluse où l'erreur d'analyse s'est produite. Utilisez le champ VALUE pour afficher la valeur des données qui ont été réellement analysées dans cette colonne, y compris les colonnes qui ont été correctement analysées jusqu'à l'erreur.

Cette vue est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_LOADERROR_DETAIL contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de

dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_LOAD_ERROR_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
slice	entier	Tranche où l'erreur s'est produite.
séance	entier	ID de séance pour la séance effectuant le chargement.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
filename	character(256)	Chemin d'accès complet du fichier d'entrée de la charge.
line_number	bigint	Numéro de ligne dans le fichier de chargement avec l'erreur.
field	entier	Champ avec l'erreur.
colname	character(1024)	Nom de la colonne.
value	character(1024)	Valeur des données analysées du champ. (Peut être tronquée.) Les caractères multioctets des données de chargement sont remplacés par un point.
is_null	entier	Indique si la valeur analysée est null ou pas.
type	character(10)	Type de données du champ.

Nom de la colonne	Type de données	Description
col_length	character(10)	Longueur de la colonne, le cas échéant. Ce champ est rempli lorsque le type de données a une longueur limite. Par exemple, pour une colonne avec le type de données « character(3) », cette colonne contient la valeur « 3 ».

Exemple de requête

La requête suivante joint STL_LOAD_ERRORS et STL_LOADERROR_DETAIL pour afficher les détails d'une erreur d'analyse qui s'est produite pendant le chargement de la table EVENT, qui a 100133 comme ID de table :

```
select d.query, d.line_number, d.value,
le.raw_line, le.err_reason
from stl_loaderror_detail d, stl_load_errors le
where
d.query = le.query
and tbl = 100133;
```

L'exemple de sortie suivant illustre les colonnes qui ont été correctement chargées, y compris la colonne avec l'erreur. Dans cet exemple, deux colonnes ont été chargées avec succès avant que l'erreur d'analyse ne se produise dans la troisième colonne, où une chaîne de caractères n'a pas été correctement analysée pour un champ destiné à un nombre entier. Comme le champ prévoyait un nombre entier, il a analysé la chaîne « aaa », qui correspond à des données non initialisées, comme valeur null et a généré une erreur d'analyse. La sortie affiche la valeur brute, la valeur analysée et la raison de l'erreur :

```
query  | line_number | value | raw_line | err_reason
-------+-------------+-------+----------+----------------
4      |      3      |  1201 |  1201    | Invalid digit
4      |      3      |   126 |   126    | Invalid digit
4      |      3      |       |   aaa    | Invalid digit
(3 rows)
```

Lorsqu'une requête joint STL_LOAD_ERRORS et STL_LOADERROR_DETAIL, elle affiche un motif d'erreur pour chaque colonne des données brutes, ce qui signifie simplement qu'une erreur s'est

produite sur cette ligne. La dernière ligne des résultats est la colonne où l'erreur d'analyse s'est produite.

STL_MERGE

Analyse les étapes d'exécution de la fusion pour les requêtes. Ces étapes se produisent lorsque les résultats d'opérations parallèles (telles que tris et jointures) sont fusionnés en vue d'un traitement ultérieur.

STL_MERGE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_MERGE contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.

Nom de la colonne	Type de données	Description
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **2009-06-12 11:29:19.131358** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **2009-06-12 11:29:19.131358** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.

Exemples de requêtes

L'exemple suivant renvoie 10 résultats d'exécution de fusion.

```
select query, step, starttime, endtime, tasknum, rows
from stl_merge
limit 10;
```

```
 query | step |       starttime     |        endtime      | tasknum | rows
-------+------+---------------------+---------------------+---------+------
     9 |    0 | 2013-08-12 20:08:14 | 2013-08-12 20:08:14 |       0 |    0
    12 |    0 | 2013-08-12 20:09:10 | 2013-08-12 20:09:10 |       0 |    0
    15 |    0 | 2013-08-12 20:10:24 | 2013-08-12 20:10:24 |       0 |    0
    20 |    0 | 2013-08-12 20:11:27 | 2013-08-12 20:11:27 |       0 |    0
    26 |    0 | 2013-08-12 20:12:28 | 2013-08-12 20:12:28 |       0 |    0
    32 |    0 | 2013-08-12 20:14:33 | 2013-08-12 20:14:33 |       0 |    0
    38 |    0 | 2013-08-12 20:16:43 | 2013-08-12 20:16:43 |       0 |    0
    44 |    0 | 2013-08-12 20:17:05 | 2013-08-12 20:17:05 |       0 |    0
    50 |    0 | 2013-08-12 20:18:48 | 2013-08-12 20:18:48 |       0 |    0
    56 |    0 | 2013-08-12 20:20:48 | 2013-08-12 20:20:48 |       0 |    0
```

```
(10 rows)
```

STL_MERGEJOIN

Analyse les étapes d'exécution de jointure de fusion pour les requêtes.

STL_MERGEJOIN est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_MERGEJOIN contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6

Nom de la colonne	Type de données	Description
		chiffres de précision pour les fractions de secondes. Par exemple : **2009-06-12 11:29:19.131358** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **2009-06-12 11:29:19.131358** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
tbl	entier	ID de table. Il s'agit de l'ID de la table interne qui a été utilisée dans la jointure de fusion.
checksum	bigint	Information à utilisation interne uniquement.

Exemples de requêtes

L'exemple suivant renvoie les résultats de jointure de fusion pour la requête la plus récente.

```
select sum(s.qtysold), e.eventname
from event e, listing l, sales s
where e.eventid=l.eventid
and l.listid= s.listid
group by e.eventname;

select * from stl_mergejoin where query=pg_last_query_id();
```

```
 userid | query | slice | segment | step |         starttime   |          endtime    |
 tasknum | rows | tbl
--------+-------+-------+---------+------+---------------------+---------------------
+---------+------+-----
    100 | 27399 |     3 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |43428 | 240
```

```
    100 | 27399 |     0 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |43159 | 240
    100 | 27399 |     2 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |42778 | 240
    100 | 27399 |     1 |       4 |    4 | 2013-10-02 16:30:41 | 2013-10-02 16:30:41 |
     19 |43091 | 240
```

STL_MV_STATE

La vue STL_MV_STATE contient une ligne pour chaque transition d'état d'une vue matérialisée.

Pour plus d'informations sur les vues matérialisées, consultez Vues matérialisées dans Amazon Redshift.

STL_MV_STATE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_MV_STATE. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	bigint	ID de l'utilisateur qui a créé l'événement.
starttime	timestamp	Heure de début de l'événement.
xid	bigint	ID de transaction de l'événement.
event_desc	char(500)	Événement ayant entraîné le changement d'état. Quelques exemples de valeur possible : • Le type de colonne a été modifié • La colonne a été supprimée • La colonne a été renommée

Nom de la colonne	Type de données	Description
		• Le nom du schéma a été modifié • Conversion de petite table • TRUNCATE • Vacuum Notez qu'il existe d'autres valeurs possibles pour cette colonne.
db_name	char(128)	Base de données contenant la vue matérialisée.
base_table_schema	char(128)	Schéma de la table de base.
base_table_name	char(128)	Nom de la table de base.
mv_schema	char(128)	Schéma de la vue matérialisée.
mv_name	char(128)	Nom de la vue matérialisée.
state	character(32)	État modifié de la vue matérialisée comme suit : • Recompute (Recalculer) • Unrefreshable (Inactualisable)

Le tableau suivant présente des exemples de combinaison `event_desc` et `state`.

```
          event_desc     |     state
-------------------------+---------------
 TRUNCATE                | Recompute
 TRUNCATE                | Recompute
 Small table conversion  | Recompute
 Vacuum                  | Recompute
 Column was renamed      | Unrefreshable
 Column was dropped      | Unrefreshable
```

```
 Table was renamed       | Unrefreshable
 Column type was changed | Unrefreshable
 Schema name was changed | Unrefreshable
```

Exemple de requête

Pour afficher le journal des transitions d'état des vues matérialisées, exécutez la requête suivante.

```
select * from stl_mv_state;
```

Cette requête renvoie l'exemple de sortie suivant :

```
 userid |         starttime          | xid  |            event_desc       | db_name |
 base_table_schema   |   base_table_name    |      mv_schema       | mv_name       |
  state
--------+----------------------------+------+-----------------------------+---------
+---------------------+---------------------+---------------------+--------------
+---------------
    138 | 2020-02-14 02:21:25.578885 | 5180 | TRUNCATE                    | dev     |
 public              | mv_base_table        | public              | mv_test       |
 Recompute
    138 | 2020-02-14 02:21:56.846774 | 5275 | Column was dropped          | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
    100 | 2020-02-13 22:09:53.041228 | 1794 | Column was renamed          | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
      1 | 2020-02-13 22:10:23.630914 | 1893 | ALTER TABLE ALTER SORTKEY   | dev     |
 public              | mv_base_table_sorted | public              | mv_test       |
 Recompute
      1 | 2020-02-17 22:57:22.497989 | 8455 | ALTER TABLE ALTER DISTSTYLE | dev     |
 public              | mv_base_table        | public              | mv_test       |
 Recompute
    173 | 2020-02-17 22:57:23.591434 | 8504 | Table was renamed           | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
    173 | 2020-02-17 22:57:27.229423 | 8592 | Column type was changed     | dev     |
                     | mv_base_table        | public              | mv_test       |
 Unrefreshable
    197 | 2020-02-17 22:59:06.212569 | 9668 | TRUNCATE                    | dev     |
 schemaf796e415850f4f | mv_base_table        | schemaf796e415850f4f | mv_test       |
 Recompute
```

```
   138 | 2020-02-14 02:21:55.705655 | 5226 | Column was renamed          | dev     |
                    | mv_base_table        | public               | mv_test       |
Unrefreshable
     1 | 2020-02-14 02:22:26.292434 | 5325 | ALTER TABLE ALTER SORTKEY   | dev     |
public               | mv_base_table_sorted | public               | mv_test       |
Recompute
```

STL_NESTLOOP

Analyse les étapes d'exécution de jointure par boucle imbriquée pour les requêtes.

STL_NESTLOOP est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_NESTLOOP contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.

Nom de la colonne	Type de données	Description
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
tbl	entier	ID de table.
checksum	bigint	Information à utilisation interne uniquement.

Exemples de requêtes

Comme la requête suivante oublie de joindre la table CATEGORY, elle produit un produit cartésien partiel, ce qui n'est pas recommandé. La requête illustre ici une boucle imbriquée.

```
select count(event.eventname), event.eventname, category.catname, date.caldate
from event, category, date
where event.dateid = date.dateid
group by event.eventname, category.catname, date.caldate;
```

La requête suivante affiche les résultats de la requête précédente dans la vue STL_NESTLOOP.

```
select query, slice, segment as seg, step,
```

```
datediff(msec, starttime, endtime) as duration, tasknum, rows, tbl
from stl_nestloop
where query = pg_last_query_id();
```

```
 query | slice | seg | step | duration | tasknum | rows  | tbl
-------+-------+-----+------+----------+---------+-------+-----
  6028 |     0 |   4 |    5 |       41 |      22 | 24277 | 240
  6028 |     1 |   4 |    5 |       26 |      23 | 24189 | 240
  6028 |     3 |   4 |    5 |       25 |      23 | 24376 | 240
  6028 |     2 |   4 |    5 |       54 |      22 | 23936 | 240
```

STL_PARSE

Analyse les étapes de requête qui analysent les chaînes en valeurs binaires pour le chargement.

STL_PARSE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_PARSE contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.

Nom de la colonne	Type de données	Description
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.

Exemples de requêtes

L'exemple suivant renvoie tous les résultats d'étape de requête pour la tranche 1 et le segment 0 où les chaînes ont été analysées en valeurs binaires.

```
select query, step, starttime, endtime, tasknum, rows
from stl_parse
where slice=1 and segment=0;
```

```
 query | step |     starttime       |        endtime      | tasknum |  rows
-------+------+---------------------+---------------------+---------+--------
   669 |    1 | 2013-08-12 22:35:13 | 2013-08-12 22:35:17 |      32 | 192497
   696 |    1 | 2013-08-12 22:35:49 | 2013-08-12 22:35:49 |      32 |      0
   525 |    1 | 2013-08-12 22:32:03 | 2013-08-12 22:32:03 |      13 |  49990
```

```
   585 |    1 | 2013-08-12 22:33:18 | 2013-08-12 22:33:19 |      13 |    202
   621 |    1 | 2013-08-12 22:34:03 | 2013-08-12 22:34:03 |      27 |    365
   651 |    1 | 2013-08-12 22:34:47 | 2013-08-12 22:34:53 |      35 | 192497
   590 |    1 | 2013-08-12 22:33:28 | 2013-08-12 22:33:28 |      19 |      0
   599 |    1 | 2013-08-12 22:33:39 | 2013-08-12 22:33:39 |      31 |     11
   675 |    1 | 2013-08-12 22:35:26 | 2013-08-12 22:35:27 |      38 |   3766
   567 |    1 | 2013-08-12 22:32:47 | 2013-08-12 22:32:48 |      23 |  49990
   630 |    1 | 2013-08-12 22:34:17 | 2013-08-12 22:34:17 |      36 |      0
   572 |    1 | 2013-08-12 22:33:04 | 2013-08-12 22:33:04 |      29 |      0
   645 |    1 | 2013-08-12 22:34:37 | 2013-08-12 22:34:38 |      29 |   8798
   604 |    1 | 2013-08-12 22:33:47 | 2013-08-12 22:33:47 |      37 |      0
(14 rows)
```

STL_PLAN_INFO

Utilisez la vue STL_PLAN_INFO pour examiner la sortie EXPLAIN d'une requête en termes d'ensemble de lignes. Il s'agit d'un autre moyen de regarder les plans de requête.

STL_PLAN_INFO est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_PLAN_INFO contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.

Nom de la colonne	Type de données	Description
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
nodeid	entier	Identificateur de nœud de plan, où un nœud correspond à une ou plusieurs étapes de l'exécution de la requête.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Numéro qui identifie l'étape de la requête.
locus	entier	Emplacement où l'étape s'exécute. 0 si sur un nœud de calcul et 1 si sur le nœud principal.
plannode	entier	Valeur énumérée du nœud de plan. Consultez le tableau suivant pour obtenir les enums pour plannode. (La colonne PLANNODE de STL_EXPLAIN contient le texte du nœud de plan.)
startupcost	double precision	Coût relatif estimé du retour de la première ligne pour cette étape.
totalcost	double precision	Coût estimé relatif de l'exécution de l'étape.
rows	bigint	Nombre estimé de lignes qui seront produites par l'étape.
octets	bigint	Nombre estimé d'octets qui seront produits par l'étape.

Exemples de requêtes

Les exemples suivants comparent les plans de requête d'une simple requête SELECT retournée à l'aide de la commande EXPLAIN et de l'interrogation de la vue STL_PLAN_INFO.

```
explain select * from category;
QUERY PLAN
------------------------------------------------------------
XN Seq Scan on category (cost=0.00..0.11 rows=11 width=49)
(1 row)
```

```
select * from category;
catid | catgroup | catname | catdesc
-------+----------+-----------+--------------------------------------------
1 | Sports | MLB | Major League Baseball
3 | Sports | NFL | National Football League
5 | Sports | MLS | Major League Soccer
...

select * from stl_plan_info where query=256;

query | nodeid | segment | step | locus | plannode | startupcost | totalcost
| rows | bytes
-------+--------+---------+------+-------+----------+-------------+-----------+------
+-------
256 | 1 | 0 | 1 | 0 | 104 | 0 | 0.11 | 11 | 539
256 | 1 | 0 | 0 | 0 | 104 | 0 | 0.11 | 11 | 539
(2 rows)
```

Dans cet exemple, PLANNODE 104 fait référence à l'analyse séquentielle de la table CATEGORY.

```
select distinct eventname from event order by 1;

eventname
----------------------------------------------------------------------
.38 Special
3 Doors Down
70s Soul Jam
A Bronx Tale
...

explain select distinct eventname from event order by 1;

QUERY PLAN
------------------------------------------------------------------------------------------
XN Merge (cost=1000000000136.38..1000000000137.82 rows=576 width=17)
Merge Key: eventname
-> XN Network (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Send to leader
-> XN Sort (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Sort Key: eventname
```

```
-> XN Unique (cost=0.00..109.98 rows=576 width=17)
-> XN Seq Scan on event (cost=0.00..87.98 rows=8798
width=17)
(8 rows)

select * from stl_plan_info where query=240 order by nodeid desc;

query | nodeid | segment | step | locus | plannode | startupcost |
totalcost | rows | bytes
-------+--------+---------+------+-------+----------+-----------------
+-----------------+------+--------
240 | 5 | 0 | 0 | 0 | 104 | 0                | 87.98   | 8798 | 149566
240 | 5 | 0 | 1 | 0 | 104 | 0                | 87.98   | 8798 | 149566
240 | 4 | 0 | 2 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 0 | 3 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 1 | 0 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 1 | 1 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 3 | 1 | 2 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 3 | 2 | 0 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 2 | 2 | 1 | 0 | 123 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 1 | 3 | 0 | 0 | 122 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
(10 rows)
```

STL_PROJECT

Contient les lignes des étapes de requête qui sont utilisées pour évaluer les expressions.

STL_PROJECT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_PROJECT contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
checksum	bigint	Information à utilisation interne uniquement.

Exemples de requêtes

L'exemple suivant renvoie toutes les lignes des étapes de requête qui ont été utilisées pour évaluer les expressions de la tranche 0 et du segment 1.

```
select query, step, starttime, endtime, tasknum, rows
from stl_project
where slice=0 and segment=1;
```

```
 query  | step |         starttime   |          endtime    | tasknum | rows
--------+------+---------------------+---------------------+---------+------
  86399 |    2 | 2013-08-29 22:01:21 | 2013-08-29 22:01:21 |      25 |   -1
  86399 |    3 | 2013-08-29 22:01:21 | 2013-08-29 22:01:21 |      25 |   -1
    719 |    1 | 2013-08-12 22:38:33 | 2013-08-12 22:38:33 |       7 |   -1
  86383 |    1 | 2013-08-29 21:58:35 | 2013-08-29 21:58:35 |       7 |   -1
    714 |    1 | 2013-08-12 22:38:17 | 2013-08-12 22:38:17 |       2 |   -1
  86375 |    1 | 2013-08-29 21:57:59 | 2013-08-29 21:57:59 |       2 |   -1
  86397 |    2 | 2013-08-29 22:01:20 | 2013-08-29 22:01:20 |      19 |   -1
    627 |    1 | 2013-08-12 22:34:13 | 2013-08-12 22:34:13 |      34 |   -1
  86326 |    2 | 2013-08-29 21:45:28 | 2013-08-29 21:45:28 |      34 |   -1
  86326 |    3 | 2013-08-29 21:45:28 | 2013-08-29 21:45:28 |      34 |   -1
  86325 |    2 | 2013-08-29 21:45:27 | 2013-08-29 21:45:27 |      28 |   -1
  86371 |    1 | 2013-08-29 21:57:42 | 2013-08-29 21:57:42 |       4 |   -1
 111100 |    2 | 2013-09-03 19:04:45 | 2013-09-03 19:04:45 |      12 |   -1
    704 |    2 | 2013-08-12 22:36:34 | 2013-08-12 22:36:34 |      37 |   -1
    649 |    2 | 2013-08-12 22:34:47 | 2013-08-12 22:34:47 |      38 |   -1
    649 |    3 | 2013-08-12 22:34:47 | 2013-08-12 22:34:47 |      38 |   -1
    632 |    2 | 2013-08-12 22:34:22 | 2013-08-12 22:34:22 |      13 |   -1
    705 |    2 | 2013-08-12 22:36:48 | 2013-08-12 22:36:49 |      13 |   -1
    705 |    3 | 2013-08-12 22:36:48 | 2013-08-12 22:36:49 |      13 |   -1
      3 |    1 | 2013-08-12 20:07:40 | 2013-08-12 20:07:40 |       3 |   -1
  86373 |    1 | 2013-08-29 21:57:58 | 2013-08-29 21:57:58 |       3 |   -1
 107976 |    1 | 2013-09-03 04:05:12 | 2013-09-03 04:05:12 |       3 |   -1
  86381 |    1 | 2013-08-29 21:58:35 | 2013-08-29 21:58:35 |       8 |   -1
  86396 |    1 | 2013-08-29 22:01:20 | 2013-08-29 22:01:20 |      15 |   -1
    711 |    1 | 2013-08-12 22:37:10 | 2013-08-12 22:37:10 |      20 |   -1
  86324 |    1 | 2013-08-29 21:45:27 | 2013-08-29 21:45:27 |      24 |   -1
(26 rows)
```

STL_QUERY

renvoie les informations d’exécution d’une requête de base de données.

> **Note**
>
> Les vues STL_QUERY et STL_QUERYTEXT contiennent uniquement des informations sur les requêtes, pas sur d’autres utilitaires ou commandes DDL. Pour obtenir la liste et

les informations de toutes les instructions exécutées par Amazon Redshift, vous pouvez également interroger les vues STL_DDLTEXT et STL_UTILITYTEXT. Pour obtenir la liste complète de toutes les instructions exécutées par Amazon Redshift vous pouvez interroger la vue SVL_STATEMENTTEXT.

STL_QUERY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Note

Pour vérifier si une transaction contenant la requête exécutée a été validée avec succès, vous devez effectuer une opération de jointure entre les tables système et la `sys_transaction_history` table. Par exemple :

```
SELECT
    stlq.xid AS transaction_id,
    stlq.query AS query_id,
    TRIM(stlq.querytxt) AS query_text,
    th.status AS transaction_status
FROM
    stl_query stlq
LEFT JOIN
    sys_transaction_history th ON stlq.xid = th.transaction_id;
```

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.

Nom de la colonne	Type de données	Description
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
étiquette	caractère (320)	Le nom du fichier utilisé pour exécuter la requête ou une étiquette définie avec une commande SET QUERY_GROUP. Si la requête n'est pas basée sur un fichier ou si le paramètre QUERY_GROUP n'est pas défini, la valeur du champ est `default`.
xid	bigint	ID de transaction.
pid	entier	ID du processus. En règle générale, toutes les requêtes d'une séance étant exécutées dans le même processus, cette valeur reste constante si vous exécutez une série de requêtes dans la même séance. Suite à certains événements internes, Amazon Redshift peut redémarrer une séance active et attribuer un nouveau PID. Pour plus d'informations, consultez STL_RESTARTED_SESSIONS.
database	character(32)	Nom de la base de données à laquelle l'utilisateur était connecté lorsque la requête a été émise.
querytxt	character(4000)	Texte réel de la requête.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .

Nom de la colonne	Type de données	Description
aborted	entier	Si une requête a été arrêtée par le système ou annulée par l'utilisateur, cette colonne contient **1**. Si la requête est terminée (y compris le retour des résultats au client), cette colonne contient **0**. Si un client se déconnecte avant de recevoir les résultats, la requête est marquée comme annulée (**1**), même si elle s'est correctement terminée dans le backend.
insert_pristine	entier	Si les requêtes d'écriture are/were peuvent être exécutées pendant l' is/was exécution de la requête en cours. 1 = aucune requête d'écriture autorisée. 0 = requêtes d'écriture autorisées. Cette colonne est destinée à être utilisée dans le débogage.
concurrency_scaling_status	entier	Indique si la requête a été exécutée sur le cluster principal ou sur un cluster de mise à l'échelle de simultanéité. Les valeurs possibles sont les suivantes : 0 - Exécutée sur le cluster principal 1 - Exécutée sur un cluster de mise à l'échelle de simultanéité Supérieur à 1 - Exécutée sur le cluster principal

Exemples de requêtes

La requête suivante répertorie les cinq requêtes les plus récentes.

```
select query, trim(querytxt) as sqlquery
from stl_query
order by query desc limit 5;

query |                                  sqlquery
------+-------------------------------------------------
129 | select query, trim(querytxt) from stl_query order by query;
```

```
128 | select node from stv_disk_read_speeds;
127 | select system_status from stv_gui_status
126 | select * from systable_topology order by slice
125 | load global dict registry
(5 rows)
```

La requête suivante renvoie la durée écoulée, par ordre décroissant, des requêtes exécutées le 15 février 2013.

```
select query, datediff(seconds, starttime, endtime),
trim(querytxt) as sqlquery
from stl_query
where starttime >= '2013-02-15 00:00' and endtime < '2013-02-16 00:00'
order by date_diff desc;

 query | date_diff |  sqlquery
-------+-----------+-----------------------------------------
 55    |       119 | padb_fetch_sample: select count(*) from category
121    |         9 | select * from svl_query_summary;
181    |         6 | select * from svl_query_summary where query in(179,178);
172    |         5 | select * from svl_query_summary where query=148;
...
(189 rows)
```

La requête suivante montre le temps d'attente et le temps d'exécution des requêtes. Les requêtes avec `concurrency_scaling_status = 1` ont été exécutées sur un cluster de mise à l'échelle de simultanéité. Toutes les autres requêtes ont été exécutées sur le cluster principal.

```
SELECT w.service_class AS queue
     , q.concurrency_scaling_status
     , COUNT( * ) AS queries
     , SUM( q.aborted )  AS aborted
     , SUM( ROUND( total_queue_time::NUMERIC / 1000000,2 ) ) AS queue_secs
     , SUM( ROUND( total_exec_time::NUMERIC / 1000000,2 ) )  AS exec_secs
FROM stl_query q
     JOIN stl_wlm_query w
          USING (userid,query)
WHERE q.userid > 1
  AND service_class > 5
  AND q.starttime > '2019-03-01 16:38:00'
  AND q.endtime   < '2019-03-01 17:40:00'
GROUP BY 1,2
```

```
ORDER BY 1,2;
```

STL_QUERY_METRICS

Contient des informations sur les métriques, telles que le nombre total de lignes traitées et d'opérations d'entrée/sortie, l'utilisation de l'UC et l'utilisation du disque pour les requêtes dont l'exécution est terminée dans les files d'attente de requêtes définies par l'utilisateur (classes de service). Pour afficher les métriques des requêtes actives qui sont en cours d'exécution, consultez la vue système STV_QUERY_METRICS.

Les métriques de requête sont échantillonnées toutes les secondes. Par conséquent, la même requête exécutée plusieurs fois peut renvoyer des heures légèrement différentes. En outre, il est possible que les segments de requête qui s'exécutent en moins d'une seconde ne soient pas enregistrés.

STL_QUERY_METRICS suit et regroupe les métriques au niveau de la requête, du segment et de l'étape. Pour plus d'informations sur les segments et les étapes de requête, consultez Workflow d'exécution et de planification de requête. De nombreuses métriques (`max_rows`, `cpu_time`, etc.) sont additionnées entre les tranches de nœuds. Pour plus d'informations sur les tranches de nœuds, consultez Architecture système de l'entrepôt des données.

Pour déterminer le niveau auquel la ligne communique les métriques, observez les colonnes `segment` et `step_type`.

- Si `segment` et `step_type` sont définis sur `-1`, la ligne communique les métriques au niveau de la requête.
- Si `segment` n'est pas défini sur `-1` et si `step_type` est défini sur `-1`, la ligne communique les métriques au niveau du segment.
- Si `segment` et `step_type` ne sont pas définis sur `-1`, la ligne communique les métriques au niveau de l'étape.

Les vues SVL_QUERY_METRICS et SVL_QUERY_METRICS_SUMMARY regroupent les données de cette vue et présentent les informations de manière plus accessible.

STL_QUERY_METRICS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a exécuté la requête qui a généré l'entrée.
service_class	entier	ID de la classe de service. Les files d'attente de requêtes sont définies dans la configuration WLM. Les métriques sont communiquées uniquement pour les files d'attente définies par l'utilisateur.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
segment	entier	Numéro de segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Les segments de requête peuvent s'exécuter en parallèle . Chaque segment s'exécute dans un processus unique. Si le segment a une valeur de -1, les valeurs du segment de métriques sont reportées au niveau de la requête.
step_type	entier	Type de l'étape exécutée. Pour obtenir une description des types d'étapes, consultez Types d'étapes.
starttime	timestamp	Heure au format UTC de début de l'exécution de la requête, avec 6 chiffres de précision pour les fractions de secondes. Par exemple : `2009-06-12 11:29:19.131358` .
slices	entier	Nombre de tranches du cluster.
max_rows	bigint	Nombre maximal de lignes produites pour une étape, regroupées entre toutes les tranches.

Nom de la colonne	Type de données	Description
rows	bigint	Nombre de lignes traitées par une étape.
max_cpu_time	bigint	Durée maximale d'utilisation de l'UC, en microsecondes. Au niveau du segment, durée maximale d'utilisation de l'UC par le segment entre toutes les tranches. Au niveau de la requête, durée maximale d'utilisation de l'UC par n'importe quel segment de requête.
cpu_time	bigint	Durée d'utilisation de l'UC, en microsecondes. Au niveau du segment, durée totale d'utilisation de l'UC pour le segment entre toutes les tranches. Au niveau de la requête, somme des durées d'utilisation de l'UC pour la requête entre toutes les tranches et tous les segments.
max_blocks_read	bigint	Nombre maximal de blocs d'1 Mo lus par le segment, regroupés entre toutes les tranches. Au niveau du segment, nombre maximal de blocs d'1 Mo lus pour le segment entre toutes les tranches. Au niveau de la requête, nombre maximal de blocs d'1 Mo lus par n'importe quel segment de la requête.
blocks_read	bigint	Nombre de blocs d'1 Mo lus par la requête ou le segment.
max_run_time	bigint	Durée maximale écoulée pour un segment (en microsecondes). Au niveau du segment, durée maximale d'exécution pour le segment entre toutes les tranches. Au niveau de la requête, durée maximale d'exécution pour n'importe quel segment de requête.

Nom de la colonne	Type de données	Description
run_time	bigint	Somme des durées totales d'exécution, entre les tranches. La durée d'attente n'est pas incluse dans le délai d'exécution. Au niveau du segment, somme des durées d'exécution pour le segment entre toutes les tranches. Au niveau de la requête, somme des durées d'exécution pour la requête entre toutes les tranches et tous les segments. Puisqu'il s'agit d'une somme, le délai d'exécution n'est pas lié au délai d'exécution de la requête.
max_blocks_to_disk	bigint	Quantité maximale d'espace disque utilisé pour écrire des résultats intermédiaires, en blocs de Mo. Au niveau du segment, quantité maximale d'espace disque utilisé par le segment entre toutes les tranches. Au niveau de la requête, quantité maximale d'espace disque utilisé par n'importe quel segment de requête.
blocks_to_disk	bigint	Quantité d'espace disque utilisé par une requête ou un segment pour écrire des résultats intermédiaires, en blocs de Mo.
étape	entier	Étape de la requête exécutée.
max_query_scan_size	bigint	Taille maximale des données analysées par une requête, en Mo. Au niveau du segment, taille maximale des données analysées par le segment entre toutes les tranches. Au niveau de la requête, taille maximale des données analysées par n'importe quel segment de requête.
query_scan_size	bigint	Taille des données analysées par une requête, en Mo.
query_priority	entier	Priorité de la requête. Les valeurs possibles sont `-1`, `0`, `1`, `2`, `3` et `4`, où `-1` signifie que cette priorité de requête n'est pas prise en charge.

Nom de la colonne	Type de données	Description
query_queue_time	bigint	Durée, en microsecondes, pendant laquelle la requête a été mise en file d'attente.
service_class_name	character (64)	Nom de la classe de service.

Exemple de requête

Pour rechercher des requêtes avec un temps UC élevé (plus de 1 000 secondes), exécutez la requête suivante.

```
Select query, cpu_time / 1000000 as cpu_seconds
from stl_query_metrics where segment = -1 and cpu_time > 1000000000
order by cpu_time;

query | cpu_seconds
------+------------
25775 |        9540
```

Pour rechercher des requêtes actives avec une jonction de boucles imbriquées qui a renvoyé plus d'un million de lignes, exécutez la requête suivante.

```
select query, rows
from stl_query_metrics
where step_type = 15 and rows > 1000000
order by rows;

query | rows
------+-----------
25775 | 2621562702
```

Pour rechercher des requêtes actives qui se sont exécutées pendant plus de 60 secondes et ont utilisé moins de 10 secondes de temps UC, exécutez la requête suivante.

```
select query, run_time/1000000 as run_time_seconds
from stl_query_metrics
```

```
where segment = -1 and run_time > 60000000 and cpu_time < 10000000;

query | run_time_seconds
------+-----------------
25775 |              114
```

STL_QUERYTEXT

Capture le texte de la requête pour les commandes SQL.

Interrogez la vue STL_QUERYTEXT afin de capturer le SQL qui a été enregistré pour les instructions suivantes :

- SELECT, SELECT INTO
- INSERT, UPDATE, DELETE
- COPY
- UNLOAD
- Les requêtes générées par l'exécution de VACUUM et ANALYZE
- CREATE TABLE AS (CTAS)

Pour interroger l'activité de ces instructions sur une période donnée, joignez les vues STL_QUERYTEXT et STL_QUERY.

> **Note**
>
> Les vues STL_QUERY et STL_QUERYTEXT contiennent uniquement des informations sur les requêtes, pas sur d'autres utilitaires ou commandes DDL. Pour obtenir la liste et les informations de toutes les instructions exécutées par Amazon Redshift, vous pouvez également interroger les vues STL_DDLTEXT et STL_UTILITYTEXT. Pour obtenir la liste complète de toutes les instructions exécutées par Amazon Redshift vous pouvez interroger la vue SVL_STATEMENTTEXT.

Consultez aussi STL_DDLTEXT, STL_UTILITYTEXT et SVL_STATEMENTTEXT.

STL_QUERYTEXT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_TEXT. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
xid	bigint	ID de transaction.
pid	entier	ID du processus. En règle générale, toutes les requêtes d'une séance étant exécutées dans le même processus , cette valeur reste constante si vous exécutez une série de requêtes dans la même séance. Suite à certains événements internes, Amazon Redshift peut redémarrer une séance active et attribuer un nouveau PID. Pour plus d'informations, consultez STL_RESTARTED_SESSIONS. Vous pouvez utiliser cette colonne pour la joindre à la vue STL_ERROR.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
sequence	entier	Lorsqu'une seule instruction contient plus de 200 caractère s, des lignes supplémentaires sont enregistrées pour l'instruction. Sequence 0 correspond à la première ligne, 1 à la deuxième, et ainsi de suite.
text	character(200)	Texte SQL, par incréments de 200 caractères. Ce champ peut contenir des caractères spéciaux tels qu'une barre oblique inverse (\\) et un caractère de saut de ligne (\n).

Exemples de requêtes

Vous pouvez utiliser la fonction PG_BACKEND_PID() pour récupérer les informations de la séance en cours. Par exemple, la requête suivante renvoie l'ID de requête et une partie du texte de la requête pour des requêtes exécutées dans la séance en cours.

```
select query, substring(text,1,60)
from stl_querytext
where pid = pg_backend_pid()
order by query desc;

 query |                          substring
-------+-----------------------------------------------------------
 28262 | select query, substring(text,1,80) from stl_querytext where
 28252 | select query, substring(path,0,80) as path from stl_unload_l
 28248 | copy category from 's3://dw-tickit/manifest/category/1030_ma
 28247 | Count rows in target table
 28245 | unload ('select * from category') to 's3://dw-tickit/manifes
 28240 | select query, substring(text,1,40) from stl_querytext where
(6 rows)
```

Reconstruction de SQL stocké

Pour reconstruire le SQL stocké dans la colonne `text` de STL_QUERYTEXT, exécutez une instruction SELECT pour créer SQL depuis une ou plusieurs parties de la colonne `text`. Avant d'exécuter le SQL reconstruit, remplacez tout caractère spécial (\n) par un saut de ligne. Le résultat de l'instruction SELECT suivante se compose de lignes de SQL reconstruit dans le champ `query_statement`.

```
SELECT query, LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END)
 WITHIN GROUP (ORDER BY sequence) as query_statement, COUNT(*) as row_count
FROM stl_querytext GROUP BY query ORDER BY query desc;
```

Par exemple, la requête suivante sélectionne 3 colonnes. La requête elle-même contient plus de 200 caractères et est stockée dans plusieurs parties de STL_QUERYTEXT.

```
select
1 AS a01234567890123456789012345678901234567890123456789012345678901234567890,
2 AS b01234567890123456789012345678901234567890123456789012345678901234567890,
3 AS b01234567890123456789012345678901234567890123
FROM stl_querytext;
```

Dans cet exemple, la requête est stockée dans plusieurs parties (lignes) de la colonne `text` de STL_DDLTEXT.

```
select query, sequence, text
from stl_querytext where query=pg_last_query_id() order by query desc, sequence limit
 10;
```

```
query  | sequence |
                        text

-------+----------
+-----------------------------------------------------------------------------------------------------
    45 |        0 | select\n1 AS
 a012345678901234567890123456789012345678901234567890123456789012345678901234567890,\n2 AS
 b012345678901234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b012345678901234567890123456789012345678901234
    45 |        1 | \nFROM stl_querytext;
```

Pour reconstruire le SQL stocké dans STL_QUERYTEXT, exécutez le SQL suivant.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS text
from stl_querytext where query=pg_last_query_id();
```

Pour utiliser le SQL reconstruit qui en résulte dans votre client, remplacez tout caractère spécial (\n) par un saut de ligne.

```
                    text

-----------------------------------------------------------------------------------------------------
 select\n1 AS a012345678901234567890123456789012345678901234567890123456789012345678901234567890,
\n2 AS b012345678901234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b012345678901234567890123456789012345678901234\nFROM stl_querytext;
```

STL_REPLACEMENTS

Affiche un journal qui enregistre quand les caractères UTF-8 non valides ont été remplacés par la commande COPY avec l'option ACCEPTINVCHARS. Une entrée de journal est ajoutée à

STL_REPLACEMENTS pour chacune des 100 premières lignes sur chaque tranche de nœud qui nécessitait au moins un remplacement.

STL_REPLACEMENTS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_NESTLOOP contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_COPY_REPLACEMENTS Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro de tranche de nœud où le remplacement s'est produit.
tbl	entier	ID de table.
starttime	timestamp	Heure de début au format UTC pour la commande COPY.
séance	entier	ID de séance pour la séance exécutant la commande COPY.

Nom de la colonne	Type de données	Description
filename	character (256)	Chemin d'accès complet vers le fichier d'entrée pour la commande COPY.
line_number	bigint	Numéro de ligne dans le fichier de données en entrée qui contenait un caractère UTF-8 non valide. Une valeur -1 indique que le numéro de ligne n'est pas disponible, par exemple, lors de la copie d'un fichier de données organisé en colonnes.
colname	character (127)	Premier champ qui contenait un caractère UTF-8 non valide.
raw_line	character (1024)	Données de chargement brutes qui contenaient un caractère UTF-8 non valide.

Exemples de requêtes

L'exemple suivant renvoie les remplacements pour l'opération COPY la plus récente.

```
select query, session, filename, line_number, colname
from stl_replacements
where query = pg_last_copy_id();

 query | session |   filename                                  | line_number | colname
 ------+---------+---------------------------------------------+-------------+--------
    96 |    6314 | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt   |         251 | city
    96 |    6314 | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt   |         317 | city
    96 |    6314 | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt   |         569 | city
    96 |    6314 | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt   |         623 | city
    96 |    6314 | s3://DOC-EXAMPLE-BUCKET/allusers_pipe.txt   |         694 | city
...
```

STL_RESTARTED_SESSIONS

Afin de maintenir une disponibilité continue après certains événements internes, Amazon Redshift peut redémarrer une séance active avec un nouvel ID de processus (PID). Lorsqu'Amazon Redshift redémarre une séance, STL_RESTARTED_SESSIONS enregistre le nouveau et l'ancien PID.

Pour plus d'informations, consultez les exemples suivants de cette section.

STL_RESTARTED_SESSIONS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_SESSION_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
currenttime	timestamp	Heure de l'événement.
dbname	character (50)	Nom de la base de données associée à la séance.
newpid	entier	ID de processus de la séance redémarrée.
oldpid	entier	ID de processus de la séance d'origine.
nom d'utilisateur	character (50)	Nom de l'utilisateur associé à la séance.
remotehost	character (45)	Nom ou adresse IP de l'hôte distant.
remoteport	character (32)	Numéro de port de l'hôte distant.
parkedtime	timestamp	Information à utilisation interne uniquement.
session_vars	character (2000)	Information à utilisation interne uniquement.

Exemples de requêtes

L'exemple suivant joint la table STL_RESTARTED_SESSIONS à la table STL_SESSIONS afin d'afficher les noms d'utilisateur pour les séances qui ont été redémarrées.

```
select process, stl_restarted_sessions.newpid, user_name
from stl_sessions
inner join stl_restarted_sessions on stl_sessions.process =
 stl_restarted_sessions.oldpid
order by process;

...
```

STL_RETURN

Contient les détails des étapes de retour des requêtes. Une étape de retour renvoie les résultats des requêtes exécutées sur les nœuds de calcul vers le nœud principal. Le nœud principal fusionne ensuite les données et renvoie les résultats au client demandeur. Pour les requêtes exécutées sur le nœud principal, une étape de retour renvoie les résultats au client.

Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Pour plus d'informations, consultez Traitement des requêtes.

STL_RETURN est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_RETURN contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
octets	bigint	Taille, en octets, de toutes les lignes de sortie de l'étape.
paquets	entier	Nombre total de paquets envoyés sur le réseau.
checksum	bigint	Information à utilisation interne uniquement.

Exemples de requêtes

La requête suivante illustre les étapes de la requête la plus récente qui ont été exécutées sur chaque tranche.

```
SELECT query, slice, segment, step, endtime, rows, packets
from stl_return where query = pg_last_query_id();

 query |  slice | segment | step |          endtime           | rows | packets
-------+--------+---------+------+----------------------------+------+---------
     4 |      2 |       3 |    2 | 2013-12-27 01:43:21.469043 |    3 |       0
     4 |      3 |       3 |    2 | 2013-12-27 01:43:21.473321 |    0 |       0
     4 |      0 |       3 |    2 | 2013-12-27 01:43:21.469118 |    2 |       0
     4 |      1 |       3 |    2 | 2013-12-27 01:43:21.474196 |    0 |       0
     4 |      4 |       3 |    2 | 2013-12-27 01:43:21.47704  |    2 |       0
     4 |      5 |       3 |    2 | 2013-12-27 01:43:21.478593 |    0 |       0
     4 |   12811|       4 |    1 | 2013-12-27 01:43:21.480755 |    0 |       0
(7 rows)
```

STL_S3CLIENT

Enregistre la durée de transfert et autres métriques de performance.

Utilisez la table STL_S3CLIENT pour trouver le temps passé à transférer des données d'Amazon S3.

STL_S3CLIENT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.

Nom de la colonne	Type de données	Description
recordtime	timestamp	Heure de l’enregistrement.
pid	entier	ID du processus. Toutes les requêtes d’une séance étant exécutées dans le même processus, cette valeur reste constante si vous exécutez une série de requêtes dans la même séance.
http_method	character (64)	Nom de la méthode HTTP correspondant à la demande Amazon S3.
bucket	character (64)	Nom du compartiment S3.
clé	character (256)	La clé correspondant à l’objet Amazon S3.
transfer_size	bigint	Nombre d’octets transférés.
data_size	bigint	Nombre d’octets de données. Cette valeur est identique à transfer_size pour les données non compressées. Si la compression a été utilisée, il s’agit de la taille des données non compressées.
start_time	bigint	Heure à laquelle le transfert a commencé (en microsecondes, depuis le 1er janvier 2000).
end_time	bigint	Heure à laquelle le transfert a fini (en microsecondes, depuis le 1er janvier 2000).
transfer_time	bigint	Temps pris par le transfert (en microsecondes).
compression_time	bigint	Partie du temps de transfert passée à décompresser les données (en microsecondes).
connect_time	bigint	Durée écoulée entre le départ et la connexion au serveur à distance (en microsecondes).

Nom de la colonne	Type de données	Description
app_connect_time	bigint	Temps écoulé entre le début et la fin du protocole SSL connect/handshake avec l'hôte distant (en microsecondes).
nouvelles tentatives	bigint	Nombre de fois où le transfert a été retenté.
request_id	char(32)	ID de requête de l'en-tête de réponse HTTP Amazon S3
extended_request_id	char(128)	ID de demande étendu provenant de la réponse d'en-tête HTTP Amazon S3 (x-amz-id-2).
ip_address	char(64)	Adresse IP du serveur (ip V4 ou V6).
is_partial	entier	Si true = 1, cette valeur indique que le fichier d'entrée est divisé en plages lors d'une opération COPY. Si false = 0, le fichier d'entrée n'est pas divisé.
start_offset	bigint	Si le fichier d'entrée est fractionné lors d'une opération COPY, cela indique la valeur de décalage du fractionnement (en octets). Si le fichier n'est pas fractionné, cette valeur est réglée sur 0.

Exemple de requête

La requête suivante renvoie le temps pris pour charger les fichiers à l'aide d'une commande COPY.

```
select slice, key, transfer_time
from stl_s3client
where query = pg_last_copy_id();
```

Résultat

```
 slice |   key                        | transfer_time
------+------------------------------+---------------
     0 | listing10M0003_part_00       |    16626716
     1 | listing10M0001_part_00       |    12894494
```

```
     2 | listing10M0002_part_00      |    14320978
     3 | listing10M0000_part_00      |    11293439
  3371 | prefix=listing10M;marker=   |       99395
```

La requête suivante convertit `start_time` et `end_time` en horodatage.

```
select userid,query,slice,pid,recordtime,start_time,end_time,
'2000-01-01'::timestamp + (start_time/1000000.0)* interval '1 second' as start_ts,
'2000-01-01'::timestamp + (end_time/1000000.0)* interval '1 second' as end_ts
from stl_s3client where query> -1 limit 5;
```

```
 userid | query | slice |  pid  |         recordtime         |   start_time    |
 end_time     |          start_ts          |           end_ts
--------+-------+-------+-------+----------------------------+-----------------
+-----------------+----------------------------+----------------------------
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.207839 | 616436837154256 |
 616436837207838 | 2019-07-14 16:27:17.154256 | 2019-07-14 16:27:17.207838
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.252521 | 616436837208208 |
 616436837252520 | 2019-07-14 16:27:17.208208 | 2019-07-14 16:27:17.25252
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.284376 | 616436837208460 |
 616436837284374 | 2019-07-14 16:27:17.20846  | 2019-07-14 16:27:17.284374
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.285307 | 616436837208980 |
 616436837285306 | 2019-07-14 16:27:17.20898  | 2019-07-14 16:27:17.285306
      0 |     0 |     0 | 23449 | 2019-07-14 16:27:17.353853 | 616436837302216 |
 616436837353851 | 2019-07-14 16:27:17.302216 | 2019-07-14 16:27:17.353851
```

STL_S3CLIENT_ERROR

Enregistre les erreurs rencontrées par une tranche lors du chargement d'un fichier à partir d'Amazon S3.

Utilisez STL_S3CLIENT_ERROR pour trouver les détails des erreurs rencontrées lors du transfert de données à partir d'Amazon S3 dans le cadre d'une commande COPY.

STL_S3CLIENT_ERROR est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues. L'ID de requête -1 est destiné à un usage interne.
sliceId	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
recordtime	timestamp	Heure de l'enregistrement.
pid	entier	ID du processus. Toutes les requêtes d'une séance étant exécutées dans le même processus, cette valeur reste constante si vous exécutez une série de requêtes dans la même séance.
http_method	character (64)	Nom de la méthode HTTP correspondant à la demande Amazon S3.
bucket	character (64)	Noms du compartiment Amazon S3.
clé	character (256)	La clé correspondant à l'objet Amazon S3.
error	character (1024)	Message d'erreur.
is_partial	entier	Si true = 1, cette valeur indique que le fichier d'entrée est divisé en plages lors d'une opération COPY. Si false = 0, le fichier d'entrée n'est pas divisé.
start_offset	bigint	Si le fichier d'entrée est fractionné lors d'une opération COPY, cela indique la valeur de décalage du fractionnement (en

Nom de la colonne	Type de données	Description
		octets). Si le fichier n'est pas fractionné, cette valeur est réglée sur 0.

Notes d'utilisation

Si plusieurs erreurs du type « connexion expirée » s'affichent, il se peut qu'il existe un problème de réseau. Si vous utilisez la fonction Routage VPC amélioré, assurez-vous qu'il existe un chemin d'accès réseau valide entre le VPC de votre cluster et vos ressources de données. Pour plus d'informations, consultez Routage VPC amélioré dans Amazon Redshift.

Exemple de requête

La requête suivante renvoie les erreurs des commandes COPY exécutées durant la séance en cours.

```
select query, sliceid, substring(key from 1 for 20) as file,
substring(error from 1 for 35) as error
from stl_s3client_error
where pid = pg_backend_pid()
order by query desc;
```

Résultat

```
 query  | sliceid |     file           |              error

--------+---------+--------------------+-----------------------------------
 362228 |      12 | part.tbl.25.159.gz | transfer closed with 1947655 bytes
 362228 |      24 | part.tbl.15.577.gz | transfer closed with 1881910 bytes
 362228 |       7 | part.tbl.22.600.gz | transfer closed with 700143 bytes r
 362228 |      22 | part.tbl.3.34.gz   | transfer closed with 2334528 bytes
 362228 |      11 | part.tbl.30.274.gz | transfer closed with 699031 bytes r
 362228 |      30 | part.tbl.5.509.gz  | Unknown SSL protocol error in conne
 361999 |      10 | part.tbl.23.305.gz | transfer closed with 698959 bytes r
 361999 |      19 | part.tbl.26.582.gz | transfer closed with 1881458 bytes
 361999 |       4 | part.tbl.15.629.gz | transfer closed with 2275907 bytes
 361999 |      20 | part.tbl.6.456.gz  | transfer closed with 692162 bytes r
(10 rows)
```

STL_SAVE

Contient les détails des étapes d'enregistrement des requêtes. Une étape d'enregistrement enregistre le flux en entrée dans une table transitoire. Une table transitoire est une table temporaire qui stocke les résultats intermédiaires pendant l'exécution des requêtes.

Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Pour plus d'informations, consultez Traitement des requêtes.

STL_SAVE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_SAVE contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.

Nom de la colonne	Type de données	Description
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
octets	bigint	Taille, en octets, de toutes les lignes de sortie de l'étape.
tbl	entier	ID de la table transitoire matérialisée.
is_diskbased	character(1)	Si cette étape de la requête a été exécutée comme une opération sur disque : true (`t`) ou false (`f`).
workmem	bigint	Nombre d'octets de mémoire de travail assignés à l'étape.

Exemples de requêtes

La requête suivante illustre les étapes d'enregistrement de la requête la plus récente qui ont été exécutées sur chaque tranche.

```
select query, slice, segment, step, tasknum, rows,  tbl
from stl_save where query = pg_last_query_id();

 query | slice | segment | step | tasknum | rows | tbl
-------+-------+---------+------+---------+------+-----
 52236 |     3 |       0 |    2 |      21 |    0 | 239
```

```
 52236 |     2 |       0 |    2 |      20 |    0 | 239
 52236 |     2 |       2 |    2 |      20 |    0 | 239
 52236 |     3 |       2 |    2 |      21 |    0 | 239
 52236 |     1 |       0 |    2 |      21 |    0 | 239
 52236 |     0 |       0 |    2 |      20 |    0 | 239
 52236 |     0 |       2 |    2 |      20 |    0 | 239
 52236 |     1 |       2 |    2 |      21 |    0 | 239
(8 rows)
```

STL_SCAN

Analyse les étapes d'analyse de table pour les requêtes. Le numéro de l'étape pour les lignes de cette table est toujours 0, car une analyse est la première étape d'un segment.

STL_SCAN est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_SCAN contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.

Nom de la colonne	Type de données	Description
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
octets	bigint	Taille, en octets, de toutes les lignes de sortie de l'étape.
fetches	bigint	Information à utilisation interne uniquement.
type	entier	ID du type de l'analyse. Pour obtenir la liste des valeurs valides, consultez le tableau suivant.
tbl	entier	ID de table.
is_rrscan	character(1)	Si la valeur est définie sur true (t), indique qu'une analyse à plage restreinte a été utilisée sur l'étape.
is_delayed_scan	character(1)	Information à utilisation interne uniquement.

Nom de la colonne	Type de données	Description
rows_pre_filter	bigint	Pour les analyses de tables permanentes, le nombre total de lignes émises avant le filtrage des lignes marquées pour la suppression (lignes fantôme) et avant l'application des filtres de requête définis par l'utilisateur.
rows_pre_user_filter	bigint	Pour les analyses de tables permanentes, le nombre total de lignes traitées après le filtrage des lignes marquées pour la suppression (lignes fantôme) mais avant l'application des filtres de requête définis par l'utilisateur.
perm_table_name	character(136)	Pour les analyses de tables permanentes, le nom de la table analysé.
is_rlf_scan	character(1)	Si la valeur est définie sur true (t), un filtrage au niveau ligne a été utilisé pour cette étape.
is_rlf_scan_reason	entier	Information à utilisation interne uniquement.
num_em_blocks	entier	Information à utilisation interne uniquement.
checksum	bigint	Information à utilisation interne uniquement.
runtime_filtering	character(1)	Si la valeur est définie sur true (t), indique que les filtres d'exécution sont appliqués.
scan_region	entier	Information à utilisation interne uniquement.
num_sortkey_as_predicate	entier	Information à utilisation interne uniquement.
row_fetcher_state	entier	Information à utilisation interne uniquement.

Nom de la colonne	Type de données	Description
consumed_scan_ranges	bigint	Information à utilisation interne uniquement.
work_stealing_reason	bigint	Information à utilisation interne uniquement.
is_vectorized_scan	character(1)	Information à utilisation interne uniquement.
is_vectorized_scan_reason	entier	Information à utilisation interne uniquement.
row_fetcher_reason	bigint	Information à utilisation interne uniquement.
topology_signature	bigint	Information à utilisation interne uniquement.
use_tpm_partition	character(1)	Information à utilisation interne uniquement.
is_rrscan_expr	character(1)	Information à utilisation interne uniquement.
scanned_mega_value	character(1)	Information à utilisation interne uniquement. Ces informations indiquent si l'étape d'analyse donnée a analysé une valeur élevée. Une valeur élevée sera stockée dans plusieurs blocs. La taille de bloc par défaut étant d'1 Mo, une valeur importante est supérieure à 1 Mo dans une configuration par défaut.

Types d'analyse

ID de type	Description
1	Données du réseau.
2	Tables utilisateur permanentes en mémoire partagée compressée.
3	Tables transitoires en lignes.
21	Chargez des fichiers depuis Amazon S3.
22	Chargez des tables à partir d'Amazon DynamoDB.
23	Chargez les données depuis une connexion SSH distante.
24	Chargez les données depuis le cluster distant (région triée). Utilisé pour le redimensionnement.
25	Chargez les données depuis le cluster distant (région non triée). Utilisé pour le redimensionnement.
28	Lisez les données à partir d'une vue de série chronologique avec UNION ALL sur plusieurs tables.
29	Lisez des données à partir de Amazon S3.
30	Lisez les informations de partition d'une table externe Amazon S3.
33	Lisez des données à partir d'une table Postgres à distance.
36	Lisez des données à partir d'une table MySQL à distance.
37	Lisez les données d'un flux Kinesis distant.

Notes d'utilisation

Idéalement, `rows` doit être relativement proche de `rows_pre_filter`. Une grande différence entre `rows` et `rows_pre_filter` indique que le moteur d'exécution analyse des lignes qui seront ignorées par la suite, ce qui est inefficace. La différence entre `rows_pre_filter` et

`rows_pre_user_filter` est le nombre de lignes fantôme de l'analyse. Exécutez une opération VACUUM pour supprimer les lignes marquées en vue de leur suppression. La différence entre `rows` et `rows_pre_user_filter` est le nombre de lignes filtrées par la requête. Si beaucoup de lignes sont rejetées par le filtre utilisateur, vérifiez votre choix de colonne de tri ou, si la raison en est due à une grande région non triée, exécutez une opération VACUUM.

Exemples de requêtes

L'exemple suivant montre que `rows_pre_filter` est supérieur à `rows_pre_user_filter` , car la table contient des lignes supprimées sur lesquelles l'opération VACUUM n'a pas été exécutée (lignes fantôme).

```
SELECT query, slice, segment,step,rows, rows_pre_filter, rows_pre_user_filter
from stl_scan where query = pg_last_query_id();

 query |  slice | segment | step | rows  | rows_pre_filter | rows_pre_user_filter
-------+--------+---------+------+-------+-----------------+----------------------
 42915 |      0 |       0 |    0 | 43159 |           86318 |                43159
 42915 |      0 |       1 |    0 |     1 |               0 |                    0
 42915 |      1 |       0 |    0 | 43091 |           86182 |                43091
 42915 |      1 |       1 |    0 |     1 |               0 |                    0
 42915 |      2 |       0 |    0 | 42778 |           85556 |                42778
 42915 |      2 |       1 |    0 |     1 |               0 |                    0
 42915 |      3 |       0 |    0 | 43428 |           86856 |                43428
 42915 |      3 |       1 |    0 |     1 |               0 |                    0
 42915 |  10000 |       2 |    0 |     4 |               0 |                    0
(9 rows)
```

STL_SCHEMA_QUOTA_VIOLATIONS

Enregistre l'occurrence, l'horodatage, XID et d'autres informations utiles lorsqu'un quota de schéma est dépassé.

STL_SCHEMA_QUOTA_VIOLATIONS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_SCHEMA_QUOTA_VIOLATIONS. Les données de la vue de surveillance SYS sont

formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
ownerid	entier	ID du propriétaire du schéma.
xid	bigint	ID de transaction associé à l'instruction.
pid	entier	ID de processus associé à l'instruction.
userid	entier	ID de l'utilisateur qui a généré l'entrée.
schema_id	entier	ID d'espace de noms ou de schéma.
nom_schéma	caractère (128)	Nom de l'espace de noms ou du schéma.
quota	entier	Quantité d'espace disque (en Mo) que le schéma peut utiliser.
disk_usage	entier	Espace disque (en Mo) actuellement utilisé par le schéma.
disk_usage_pct	double precision	Pourcentage d'espace disque actuellement utilisé par le schéma hors du quota configuré.
timestamp	horodatage sans fuseau horaire	Heure à laquelle la violation s'est produite.

Exemples de requêtes

La requête suivante montre le résultat d'une violation de quota :

```
SELECT userid, TRIM(SCHEMA_NAME) "schema_name", quota, disk_usage, disk_usage_pct,
 timestamp FROM
stl_schema_quota_violations WHERE SCHEMA_NAME = 'sales_schema' ORDER BY timestamp DESC;
```

Cette requête renvoie l'exemple de sortie suivant pour le schéma spécifié :

```
userid | schema_name  | quota | disk_usage | disk_usage_pct |timestamp
-------+--------------+-------+------------+----------------
+----------------------------
104    | sales_schema | 2048  | 2798       |  136.62        | 2020-04-20
 20:09:25.494723
(1 row)
```

STL_SESSIONS

renvoie des informations sur l'historique de la séance utilisateur.

STL_SESSIONS diffère de STV_SESSIONS en ce sens que STL_SESSIONS contient l'historique des séances, quand STV_SESSIONS contient les séances actives actuelles.

STL_SESSIONS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_SESSION_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
starttime	timestamp	Heure au format UTC du démarrage de la séance.
endtime	timestamp	Heure au format UTC de la fin de la séance.
process	entier	ID de processus de la séance.

Nom de la colonne	Type de données	Description
user_name	character(50)	Nom d'utilisateur associé à la séance.
db_name	character(50)	Nom de la base de données associée à la séance.
timeout_sec	int	La durée maximale d'inactivité d'une séance avant son expiration (exprimée en secondes). 0 indique qu'aucun délai d'attente n'a été défini.
timed_out	int	Une valeur qui indique pourquoi la connexion a été interrompue. Il peut prendre les valeurs suivantes : • 0: La connexion a été interrompue en raison d'une erreur inconnue. • 1: Le délai de connexion a expiré. • 2: Le côté client a mis fin à la connexion. • 3: une erreur interne du backend Amazon Redshift a interrompu la connexion.

Exemples de requêtes

Pour afficher l'historique des séances de la base de données TICKIT, tapez la requête suivante :

```
select starttime, process, user_name, timeout_sec, timed_out
from stl_sessions
where db_name='tickit' order by starttime;
```

Cette requête renvoie l'exemple de sortie suivant :

```
    starttime              | process |  user_name             | timeout_sec | timed_out
---------------------------+---------+------------------------+-------------
+-------------
2008-09-15 09:54:06.746705 |   32358 | dwuser                 | 120         | 1
```

```
2008-09-15 09:56:34.30275  |   32744 | dwuser                | 60          | 1
2008-09-15 11:20:34.694837 |   14906 | dwuser                | 0           | 0
2008-09-15 11:22:16.749818 |   15148 | dwuser                | 0           | 0
2008-09-15 14:32:44.66112  |   14031 | dwuser                | 0           | 0
2008-09-15 14:56:30.22161  |   18380 | dwuser                | 0           | 0
2008-09-15 15:28:32.509354 |   24344 | dwuser                | 0           | 0
2008-09-15 16:01:00.557326 |   30153 | dwuser                | 120         | 1
2008-09-15 17:28:21.419858 |   12805 | dwuser                | 0           | 0
2008-09-15 20:58:37.601937 |   14951 | dwuser                | 60          | 1
2008-09-16 11:12:30.960564 |   27437 | dwuser                | 60          | 1
2008-09-16 14:11:37.639092 |   23790 | dwuser                | 3600        | 1
2008-09-16 15:13:46.02195  |    1355 | dwuser                | 120         | 1
2008-09-16 15:22:36.515106 |    2878 | dwuser                | 120         | 1
2008-09-16 15:44:39.194579 |    6470 | dwuser                | 120         | 1
2008-09-16 16:50:27.02138  |   17254 | dwuser                | 120         | 1
2008-09-17 12:05:02.157208 |    8439 | dwuser                | 3600        | 0
(17 rows)
```

STL_SORT

Affiche les étapes d'exécution du tri pour les requêtes, telles que les étapes qui utilisent un traitement ORDER BY.

STL_SORT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_SORT contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
octets	bigint	Taille, en octets, de toutes les lignes de sortie de l'étape.
tbl	entier	ID de table.
is_diskbased	character(1)	Si la valeur est true (t), la requête a été exécutée en tant qu'opération sur disque. Si elle est false (f), la requête a été exécutée en mémoire.

Nom de la colonne	Type de données	Description
workmem	bigint	Nombre total d'octets en mémoire de travail qui ont été assignés à l'étape.
checksum	bigint	Information à utilisation interne uniquement.

Exemples de requêtes

L'exemple suivant renvoie les résultats de tri pour la tranche 0 et le segment 1.

```
select query, bytes, tbl, is_diskbased, workmem
from stl_sort
where slice=0 and segment=1;
```

```
 query |  bytes  | tbl | is_diskbased |  workmem
-------+---------+-----+--------------+-----------
   567 | 3126968 | 241 | f            | 383385600
   604 |    5292 | 242 | f            | 383385600
   675 |  104776 | 251 | f            | 383385600
   525 | 3126968 | 251 | f            | 383385600
   585 |    5068 | 241 | f            | 383385600
   630 |  204808 | 266 | f            | 383385600
   704 |       0 | 242 | f            |         0
   669 | 4606416 | 241 | f            | 383385600
   696 |  104776 | 241 | f            | 383385600
   651 | 4606416 | 254 | f            | 383385600
   632 |       0 | 256 | f            |         0
   599 |     396 | 241 | f            | 383385600
 86397 |       0 | 242 | f            |         0
   621 |    5292 | 241 | f            | 383385600
 86325 |       0 | 242 | f            |         0
   572 |    5068 | 242 | f            | 383385600
   645 |  204808 | 241 | f            | 383385600
   590 |     396 | 242 | f            | 383385600
(18 rows)
```

STL_SSHCLIENT_ERROR

Enregistre toutes les erreurs rencontrées par le client SSH.

STL_SSHCLIENT_ERROR est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
recordtime	timestamp	Heure à laquelle l'erreur a été enregistrée.
pid	entier	Processus qui a enregistré l'erreur.
ssh_username	character (1024)	Nom d'utilisateur SSH.
point de terminaison	character (1024)	Point de terminaison SSH.
command	character (4096)	Commande SSH complète.
error	character (1024)	Message d'erreur.

STL_STREAM_SEGS

Affiche les relations entre les flux et les segments simultanés.

Dans ce contexte, les streams sont des flux Amazon Redshift. Cette vue du système ne concerne pas Diffusion de l'ingestion vers une vue matérialisée.

STL_STREAM_SEGS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_STREAM_SEGS contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
stream	entier	Ensemble des segments simultanés d'une requête.
segment	entier	Numéro qui identifie le segment de requête.

Exemples de requêtes

Pour afficher la relation entre les flux et les segments simultanés de la requête la plus récente, tapez la requête suivante :

```
select *
from stl_stream_segs
where query = pg_last_query_id();
```

```
 query | stream | segment
-------+--------+---------
    10 |      1 |       2
    10 |      0 |       0
    10 |      2 |       4
    10 |      1 |       3
    10 |      0 |       1
(5 rows)
```

STL_TR_CONFLICT

Affiche les informations pour identifier et résoudre les conflits de transaction avec les tables de base de données.

Un conflit de transaction se produit lorsqu'un ou plusieurs utilisateurs interrogent et modifient des lignes de données de tables de telle sorte que leurs transactions ne puissent pas être sérialisées. La transaction qui exécute une instruction qui interromprait la mise en série est arrêtée et annulée. Chaque fois qu'un conflit de transaction se produit, Amazon Redshift écrit une ligne de données dans la table système STL_TR_CONFLICT contenant les détails sur la transaction annulée. Pour de plus amples informations, veuillez consulter Isolement sérialisable.

STL_TR_CONFLICT n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_TRANSACTION_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
xact_id	bigint	ID de transaction de la transaction annulée.
process_id	bigint	Processus associé à la transaction annulée.
xact_start_ts	timestamp	Horodatage (UTC) quand la transaction a commencé.

Nom de la colonne	Type de données	Description
abort_time	timestamp	Horodatage (UTC) quand la transaction a été arrêtée.
table_id	bigint	ID de table de la table où le conflit s'est produit.

Exemple de requête

Pour renvoyer des informations sur les conflits qui impliquaient une table particulière, exécutez une requête qui spécifie l'ID de table :

```
select * from stl_tr_conflict where table_id=100234
order by xact_start_ts;

xact_id|process_|      xact_start_ts       |        abort_time        |table_
       |id      |                          |                          |id
-------+--------+--------------------------+--------------------------+------
  1876 |  8551  |2010-03-30 09:19:15.852326|2010-03-30 09:20:17.582499|100234
  1928 | 15034  |2010-03-30 13:20:00.636045|2010-03-30 13:20:47.766817|100234
  1991 | 23753  |2010-04-01 13:05:01.220059|2010-04-01 13:06:06.94098 |100234
  2002 | 23679  |2010-04-01 13:17:05.173473|2010-04-01 13:18:27.898655|100234
(4 rows)
```

Vous pouvez obtenir l'ID de table dans la section DETAIL du message d'erreur sur les violation de sérialisation (erreur 1023).

STL_UNDONE

Affiche les informations sur les transactions qui ont été annulées.

STL_UNDONE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_TRANSACTION_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
xact_id	bigint	ID de la transaction d'annulation.
xact_id_undone	bigint	ID de la transaction qui a été annulée.
undo_start_ts	timestamp	Heure de début de la transaction d'annulation.
undo_end_ts	timestamp	Heure de fin de la transaction d'annulation.
table_id	bigint	ID de la table qui a été affectée par la transaction d'annulation.

Exemple de requête

Pour afficher un journal concis de toutes les transactions annulées, tapez la commande suivante :

```
select xact_id, xact_id_undone, table_id from stl_undone;
```

Cette commande renvoie l'exemple de sortie suivant :

```
 xact_id | xact_id_undone | table_id
---------+----------------+----------
1344 |           1344 |   100192
1326 |           1326 |   100192
1551 |           1551 |   100192
(3 rows)
```

STL_UNIQUE

Analyse les étapes de l'exécution qui se produisent quand une fonction DISTINCT est utilisée dans la liste SELECT ou lorsque des doublons sont supprimés dans une requête UNION ou INTERSECT.

STL_UNIQUE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_UNIQUE contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution

Nom de la colonne	Type de données	Description
		avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
`type`	character(6)	Type d'étape. Les valeurs valides sont : • HASHED. Indique que l'étape a utilisé l'agrégation groupée, non triée. • PLAIN. Indique que l'étape a utilisé l'agrégation non groupée, scalaire. • SORTED. Indique que l'étape a utilisé l'agrégation groupée, triée.
is_diskbased	character(1)	Si la valeur est true (t), la requête a été exécutée en tant qu'opération sur disque. Si elle est false (f), la requête a été exécutée en mémoire.
slots	entier	Nombre total de compartiments de hachage.
workmem	bigint	Nombre total d'octets en mémoire de travail qui ont été assignés à l'étape.
max_buffers_used	bigint	Nombre maximal de tampons utilisés dans la table de hachage avant d'accéder au disque.
redimensionne	entier	Information à utilisation interne uniquement.
occupied	entier	Information à utilisation interne uniquement.
flushable	entier	Information à utilisation interne uniquement.

Nom de la colonne	Type de données	Description
précharge ment _unique utilisé	character(1)	Information à utilisation interne uniquement.
octets	biginit	Nombre d'octets de toutes les lignes de sortie de l'étape.

Exemples de requêtes

Supposons que vous exécutiez la requête suivante :

```
select distinct eventname
from event order by 1;
```

En supposant que l'ID de la requête précédente soit 6313, l'exemple suivant présente le nombre de lignes générées par l'unique étape pour chaque tranche des segments 0 et 1.

```
select query, slice, segment, step, datediff(msec, starttime, endtime) as msec,
 tasknum, rows
from stl_unique where query = 6313
order by query desc, slice, segment, step;
```

```
 query | slice | segment | step | msec | tasknum | rows
-------+-------+---------+------+------+---------+------
  6313 |     0 |       0 |    2 |    0 |      22 |  550
  6313 |     0 |       1 |    1 |  256 |      20 |  145
  6313 |     1 |       0 |    2 |    1 |      23 |  540
  6313 |     1 |       1 |    1 |   42 |      21 |  127
  6313 |     2 |       0 |    2 |    1 |      22 |  540
  6313 |     2 |       1 |    1 |  255 |      20 |  158
  6313 |     3 |       0 |    2 |    1 |      23 |  542
  6313 |     3 |       1 |    1 |   38 |      21 |  146
(8 rows)
```

STL_UNLOAD_LOG

Enregistre les détails d'une opération de déchargement.

STL_UNLOAD_LOG enregistre une ligne pour chaque fichier créé par une instruction UNLOAD. Par exemple, si une instruction UNLOAD crée 12 fichiers, STL_UNLOAD_LOG contient 12 lignes correspondantes.

STL_UNLOAD_LOG est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_UNLOAD_LOG contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS et. SYS_UNLOAD_HISTORY SYS_UNLOAD_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
pid	entier	ID de processus associé à l'instruction de la requête.
path	character(1280)	Chemin d'objet Amazon S3 complet du fichier.
start_time	timestamp	Heure de début de la transaction.
end_time	timestamp	Heure de fin de la transaction.

Nom de la colonne	Type de données	Description
line_count	bigint	Nombre de lignes déchargées dans le fichier.
transfer_size	bigint	Nombre d'octets transférés.
file_format	character(10)	Nombre de fichiers déchargés

Exemple de requête

Pour obtenir la liste des fichiers qui ont été écrits dans Amazon S3 par une commande UNLOAD, vous pouvez appeler une opération de liste Amazon S3 une fois l'action UNLOAD terminée. Vous pouvez également interroger STL_UNLO_LOG.

La requête suivante renvoie le chemin d'accès pour les fichiers créés par une instruction UNLOAD pour la dernière requête exécutée :

```
select query, substring(path,0,40) as path
from stl_unload_log
where query = pg_last_query_id()
order by path;
```

Cette commande renvoie l'exemple de sortie suivant :

```
 query |             path
-------+---------------------------------------
  2320 | s3://amzn-s3-demo-bucket/venue0000_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0001_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0002_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0003_part_00
(4 rows)
```

STL_USAGE_CONTROL

La vue STL_USAGE_CONTROL contient des informations consignées lorsqu'une limite d'utilisation est atteinte. Pour de plus amples informations sur les limites d'utilisation, consultez Gestion des limites d'utilisation du Guide de gestion Amazon Redshift.

STL_USAGE_CONTROL n'est visible que par les super-utilisateurs. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
eventtime	timestamp	Heure (UTC) pendant laquelle la requête a dépassé une limite d'utilisation.
query	entier	Identificateur de requête. Vous pouvez utiliser cet ID pour joindre d'autres vues et tables système.
xid	bigint	Identificateur de transaction.
pid	entier	Identificateur de processus associé à la requête.
usage_limit_id	character(40)	Identificateur universel unique (UUID) généré par Amazon Redshift, par exemple, `25d9297e-3e7b-41c8-9f4d-c4b6eb731c09` .
feature_type	character(30)	Fonctionnalité dont la limite d'utilisation a été dépassée. Les valeurs possibles incluent `CONCURRENCY_SCALING` et `SPECTRUM`.

Exemple de requête

L'exemple SQL suivant renvoie une partie des informations consignées lorsqu'une limite d'utilisation est atteinte.

```
select query, pid, eventtime, feature_type
from stl_usage_control
order by eventtime desc
limit 5;
```

STL_USERLOG

Enregistre les détails des modifications suivantes apportées à un utilisateur de base de données :

- Créer un utilisateur
- Supprimer un utilisateur
- Modifier un utilisateur (renommer)
- Modifier un utilisateur (modifier les propriétés)

STL_USERLOG n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_USERLOG. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur affecté par la modification.
nom d'utilisateur	character(50)	Nom d'utilisateur de l'utilisateur affecté par la modification.
olduserna me	character(50)	Pour une action d'attribution d'un nouveau nom, le nom original de l'utilisateur. Pour toute autre action, ce champ est vide.
action	character(10)	Action qui s'est produite. Valeurs valides : • Alter • Création • Drop • Rename
usecreate db	entier	Si true (1), indique que l'utilisateur a créé des privilèges de base de données.
usesuper	entier	Si true (1), indique que l'utilisateur est un super-utilisateur.

Nom de la colonne	Type de données	Description
usecatupd	entier	Si true (1), indique que l'utilisateur peut mettre à jour les catalogues système.
valuntil	timestamp	Date d'expiration du mot de passe.
pid	entier	ID du processus.
xid	bigint	ID de transaction.
recordtime	timestamp	Heure au format UTC du début de la requête.

Exemples de requêtes

L'exemple suivant exécute quatre actions utilisateur, puis interroge la vue STL_USERLOG.

```
create user userlog1 password 'Userlog1';
alter user userlog1 createdb createuser;
alter user userlog1 rename  to userlog2;
drop user userlog2;

select userid, username, oldusername, action, usecreatedb, usesuper from stl_userlog
 order by recordtime desc;
```

```
 userid |  username | oldusername |  action | usecreatedb | usesuper
--------+-----------+-------------+---------+-------------+----------
    108 | userlog2  |             | drop    |           1 |   1
    108 | userlog2  | userlog1    | rename  |           1 |   1
    108 | userlog1  |             | alter   |           1 |   1
    108 | userlog1  |             | create  |           0 |   0
 (4 rows)
```

STL_UTILITYTEXT

Capture le texte des commandes SQL autres que SELECT exécutées sur la base de données.

Interrogez la vue STL_UTILITYTEXT pour capturer le sous-ensemble suivant d'instructions SQL exécutées sur le système :

- ABORT, BEGIN, COMMIT, END, ROLLBACK
- ANALYSE
- CALL
- ANNULER
- COMMENT
- CREATE, ALTER, DROP DATABASE
- CREATE, ALTER, DROP USER
- EXPLAIN
- GRANT, REVOKE
- LOCK
- RESET
- SET
- MONTRER
- TRUNCATE

Consultez aussi STL_DDLTEXT, STL_QUERYTEXT et SVL_STATEMENTTEXT.

Utilisez les colonnes STARTTIME et ENDTIME pour découvrir quelles instructions ont été enregistrées pendant une période donnée. Les longs blocs de texte SQL sont divisés en lignes de 200 caractères de long ; la colonne SEQUENCE identifie les fragments de texte qui appartiennent à une seule instruction.

STL__UTILITYTEXT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être

plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
xid	bigint	ID de transaction.
pid	entier	ID de processus associé à l'instruction de la requête.
étiquette	caractère (320)	Le nom du fichier utilisé pour exécuter la requête ou une étiquette définie avec une commande SET QUERY_GRO UP. Si la requête n'est pas basée sur un fichier ou si le paramètre QUERY_GROUP n'est pas défini, le champ est vide.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
sequence	entier	Lorsqu'une seule instruction contient plus de 200 caractère s, des lignes supplémentaires sont enregistrées pour l'instruction. Sequence 0 correspond à la première ligne, 1 à la deuxième, et ainsi de suite.
text	character(200)	Texte SQL, par incréments de 200 caractères. Ce champ peut contenir des caractères spéciaux tels qu'une barre oblique inverse (\\) et un caractère de saut de ligne (\n).

Exemples de requêtes

La requête suivante renvoie le texte des commandes « utility » exécutées le 26 janvier 2012. Dans ce cas, quelques commandes SET et une commande SHOW ALL ont été exécutées :

```
select starttime, sequence, rtrim(text)
from stl_utilitytext
where starttime like '2012-01-26%'
order by starttime, sequence;

starttime          | sequence |              rtrim
---------------------------+-----+---------------------------------
2012-01-26 13:05:52.529235 |   0 | show all;
2012-01-26 13:20:31.660255 |   0 | SET query_group to ''
2012-01-26 13:20:54.956131 |   0 | SET query_group to 'soldunsold.sql'
...
```

Reconstruction de SQL stocké

Pour reconstruire le SQL stocké dans la colonne `text` de STL_UTILITYTEXT, exécutez une instruction SELECT pour créer SQL depuis une ou plusieurs parties de la colonne `text`. Avant d'exécuter le SQL reconstruit, remplacez tout caractère spécial (\n) par un saut de ligne. Le résultat de l'instruction SELECT suivante se compose de lignes de SQL reconstruit dans le champ `query_statement`.

```
SELECT LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END) WITHIN
 GROUP (ORDER BY sequence) as query_statement
FROM stl_utilitytext GROUP BY xid order by xid;
```

Par exemple, la requête suivante définit une chaîne de zéros pour query_group. La requête elle-même contient plus de 200 caractères et est stockée dans plusieurs parties de STL_UTILITYTEXT.

```
set query_group to
 '00000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
             000000';
```

Dans cet exemple, la requête est stockée dans deux parties (lignes) de la colonne `text` de STL_UTILITYTEXT.

```
select query, sequence, text
```

```
from stl_utilitytext where query=pg_last_query_id() order by query desc, sequence limit
 10;
```

```
         starttime          | sequence |
                                                    text

----------------------------+----------
+-----------------------------------------------------------------------------------------------
 2019-07-23 22:55:34.926198 |        0 | set query_group to
 '00000000000000000000000000000000000000000000000000000000000000000000000000000000\n00000000000
 2019-07-23 22:55:34.926198 |        1 |                   000000';
```

Pour reconstruire le SQL stocké dans STL_UTILITYTEXT, exécutez le SQL suivant.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS query_statement
from stl_utilitytext where query=pg_last_query_id();
```

Pour utiliser le SQL reconstruit qui en résulte dans votre client, remplacez tout caractère spécial (\n) par un saut de ligne.

```
                                                  query_statement


-----------------------------------------------------------------------------------------------
 set query_group to
 '00000000000000000000000000000000000000000000000000000000000000000000000000000000\n00000000000
                  000000';
```

STL_VACUUM

Affiche les statistiques de ligne et de bloc pour les tables qui ont été l'objet d'une opération VACUUM.

La vue affiche les informations propres à chaque moment où l'opération VACUUM a commencé et fini, et démontre les avantages de l'exécution de l'opération. Pour plus d'informations sur la

configuration requise pour exécuter cette commande, consultez la description de la commande VACUUM.

STL_VACUUM n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_VACUUM_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
xid	bigint	ID de transaction de l'instruction VACUUM. Vous pouvez joindre cette table à la vue STL_QUERY pour afficher les instructions qui ont été exécutées pour une transacti on VACUUM donnée. Si vous exécutez cette opération sur l'ensemble de la base de données, chaque table est aspirée dans une transaction distincte.
table_id	entier	ID de la table.
status	character(30)	État de l'opération VACUUM pour chaque table. Les valeurs possibles sont les suivantes : • **`Started`** • **`Started Delete Only`** • **`Started Delete Only (Sorted >= nn%)`** Seule la phase de suppression a été démarrée pour une opération VACUUM FULL. La phase de tri a été ignorée, car la table a été déjà été triée au niveau du seuil de tri ou au-dessus. • **`Started Sort Only`**

Nom de la colonne	Type de données	Description
		• **Started Ranged Partition** • **Started Reindex** • **Finished** Heure à laquelle l'opération s'est terminée pour la table. Pour découvrir combien de temps une opération VACUUM a nécessité sur une table spécifique, soustrayez l'heure de début de l'heure de fin pour un ID de transaction et un ID de table donnés. • **Skipped** La table a été ignorée, car elle a été entièrement triée et aucune ligne n'a été marquée en vue de sa suppression. • **Skipped (delete only)** La table a été ignorée parce que DELETE ONLY a été spécifié et qu'aucune ligne n'a été marquée en vue de sa suppression. • **Skipped (sort only)** La table a été ignorée, car SORT ONLY a été spécifié et que la table a déjà été entièrement triée. • **Skipped (sort only, sorted>=xx%)** La table a été ignorée, car SORT ONLY a été spécifié et que la table a été déjà triée au niveau ou au-dessus du seuil de tri. • **Skipped (0 rows)** La table a été ignorée, car elle est vide. • **VacuumBG**

Nom de la colonne	Type de données	Description
		Une opération VACUUM automatique a été exécutée en arrière-plan. Cet état est ajouté aux autres états lorsqu'ils sont exécutés automatiquement. Par exemple, une opération VACUUM DELETE ONLY exécutée automatiquement présenterait une ligne de départ avec l'état `[VacuumBG] Started Delete Only`. Pour plus d'informations sur le réglage du seuil de tri de VACUUM, consultez VACUUM.
rows	bigint	Nombre réel de lignes de la table, plus les lignes supprimées qui sont toujours stockées sur disque (en attente d'une opération VACUUM). Cette colonne affiche le nombre de lignes, avant le début de VACUUM, avec un état **`Started`** et le nombre de lignes, à la fin de VACUUM, avec un état **`Finished`**.
sortedrows	entier	Nombre de lignes de la table qui sont triées. Cette colonne affiche le nombre de lignes, avant le début de VACUUM, avec **`Started`** dans la colonne d'état, et le nombre de lignes, à la fin de VACUUM, avec **`Finished`** dans la colonne d'état.
Blocs de	entier	Nombre total de blocs de données utilisés pour stocker les données de la table avant l'opération VACUUM (lignes avec un état **`Started`**) et après l'opération (colonne **`Finished`**). Chaque bloc de données utilise 1 Mo.

Nom de la colonne	Type de données	Description
max_merge_partitions	entier	Cette colonne est utilisée pour l'analyse des performances et représente le nombre maximal de partitions qu'une opération VACUUM peut traiter pour la table par itération de phase de fusion. (L'opération VACUUM trie la région non triée en une ou plusieurs partitions. Selon le nombre de colonnes de la table et la configuration actuelle d'Amazon Redshift, la phase de fusion peut traiter un nombre maximal de partitions en une seule itération de fusion. La phase de fusion continue si le nombre de partitions triées dépasse le nombre maximal de partitions de fusion, mais un plus grand nombre d'itérations de fusion sera nécessaire.)
eventtime	timestamp	Heure de début ou de fin de l'opération VACUUM.
reclaimable_rows	bigint	Nombre de lignes récupérables pour le cutoff_xid actuel. Cette colonne indique le nombre estimé de lignes récupérables de Redshift avant le début de l'opération VACUUM pour les lignes ayant un état **`Started`**, et le nombre réel de lignes récupérables restantes après l'opération VACUUM pour les lignes ayant un état **`Finished`**.
reclaimable_space_mb	bigint	Espace récupérable en Mo pour le cutoff_xid actuel. Cette colonne indique la quantité d'espace récupérable estimée de Redshift avant le démarrage de l'opération VACUUM pour les lignes ayant un état **`Started`**, et la quantité réelle d'espace récupérable restant après l'opération VACUUM pour les lignes ayant un état **`Finished`**.
cutoff_xid	bigint	ID de transaction de coupure (« cutoff ») de l'opération VACUUM. Les transactions postérieures à la coupure ne sont pas incluses dans l'opération VACUUM.

Nom de la colonne	Type de données	Description
is_recluster	entier	Si la valeur est 1 (vrai), l'opération VACUUM a exécuté l'algorithme RECLUSTER, si la valeur est 0 (faux), cela n'a pas été le cas.

Exemples de requêtes

La requête suivante rapporte les statistiques de l'opération VACUUM pour la table 108313. La table a été l'objet d'une opération VACUUM après une série d'insertions et de suppressions.

```
select xid, table_id, status, rows, sortedrows, blocks, eventtime,
                reclaimable_rows, reclaimable_space_mb
from stl_vacuum where table_id=108313 order by eventtime;

 xid   | table_id | status                  | rows | sortedrows | blocks |  eventtime
         | reclaimable_rows | reclaimable_space_mb
-------+----------+-------------------------+------+------------+--------
+----------------------+------------------+----------------------
 14294 |   108313 | Started                 | 1950 |        408 |     28 |  2016-05-19
 17:36:01 |              984 |                   17
 14294 |   108313 | Finished                |  966 |        966 |     11 |  2016-05-19
 18:26:13 |                0 |                    0
 15126 |   108313 | Skipped(sorted>=95%)    |  966 |        966 |     11 |  2016-05-19
 18:26:38 |                0 |                    0
```

Au début de l'opération VACUUM, la table contenait 1 950 lignes stockées dans 28 blocs de 1 Mo. Amazon Redshift a estimé pouvoir en récupérer 984, soit 17 blocs d'espace disque, avec une opération VACUUM.

Dans la ligne correspondant à l'état Finished (Terminé), la colonne ROWS présente une valeur de 966, et la valeur de la colonne BLOCKS est égale à 11, au lieu de 28. L'opération VACUUM a récupéré la quantité d'espace disque estimée, sans qu'il ne reste de lignes ou d'espace récupérables à l'issue de l'opération VACUUM.

Dans la phase de tri (transaction 15126), l'opération VACUUM a été en mesure d'ignorer la table, car les lignes ont été ajoutées dans l'ordre des clés de tri.

L'exemple suivant montre les statistiques d'une opération d'aspiration SORT ONLY sur la table SALES (110116 dans cet exemple) après une importante opération INSERT :

```
vacuum sort only sales;

select xid, table_id, status, rows, sortedrows, blocks, eventtime
from stl_vacuum order by xid, table_id, eventtime;

xid |table_id|     status      | rows  |sortedrows|blocks|     eventtime
----+--------+-----------------+-------+----------+------+--------------------
...
2925| 110116 |Started Sort Only|1379648|   172456 |  132 | 2011-02-24 16:25:21...
2925| 110116 |Finished         |1379648|  1379648 |  132 | 2011-02-24 16:26:28...
```

STL_WINDOW

Analyse les étapes de requête qui effectuent les fonctions de fenêtrage.

STL_WINDOW est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STL_WINDOW contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.

Nom de la colonne	Type de données	Description
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
slice	entier	Numéro identifiant la tranche au cours de laquelle la requête était en cours d'exécution.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure au format UTC de l'exécution de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
tasknum	entier	Numéro du processus de la tâche de requête qui a été assigné pour exécuter l'étape.
rows	bigint	Nombre total de lignes traitées.
is_diskbased	character(1)	Si la valeur est true (t), la requête a été exécutée en tant qu'opération sur disque. Si elle est false (f), la requête a été exécutée en mémoire.
workmem	bigint	Nombre total d'octets en mémoire de travail qui ont été assignés à l'étape.

Exemples de requêtes

L'exemple suivant renvoie les résultats de la fonction fenêtrage pour la tranche 0 et le segment 3.

```
select query, tasknum, rows, is_diskbased, workmem
from stl_window
where slice=0 and segment=3;
```

```
 query | tasknum | rows | is_diskbased | workmem
-------+---------+------+--------------+----------
 86326 |      36 | 1857 | f            | 95256616
   705 |      15 | 1857 | f            | 95256616
 86399 |      27 | 1857 | f            | 95256616
   649 |      10 |    0 | f            | 95256616
(4 rows)
```

STL_WLM_ERROR

Enregistre toutes les erreurs liées à WLM lorsqu'elles se produisent.

STL_WLM_ERROR est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
recordtime	timestamp	Heure à laquelle l'erreur s'est produite.
pid	entier	ID du processus qui a généré l'erreur.
error_string	character(256)	Description de l'erreur.

STL_WLM_RULE_ACTION

Enregistre des détails relatifs aux actions résultant des règles de surveillance de requête WLM associées aux files d'attente définies par l'utilisateur. Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.

STL_WLM_RULE_ACTION est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	Utilisateur qui a exécuté la requête.
query	entier	ID de requête.
service_class	entier	ID de la classe de service. Les files d'attente de requêtes sont définies dans la configuration WLM. Les classes de service supérieures à 5 sont des files d'attente définies par l'utilisateur.
règle	character(256)	Nom d'une règle de surveillance de requête.
action	character(256)	Action obtenue. Les valeurs possibles sont les suivantes : • journal • hop(reassign) • hop(restart) • annuler • change_query_priority • none La valeur none indique que les prédicats de la règle ont été respectés, mais que l'action a été remplacée par une autre règle comprenant une action avec une gravité plus élevée.
recordtime	timestamp	Heure à laquelle l'action a été enregistrée en UTC.
action_value	character(256)	Si `action` est `change_query_priority`, les valeurs possibles sont `highest`, `high`, `normal`, `low` et `lowest`.

Nom de la colonne	Type de données	Description
		Si `action` est `log`, `hop` ou `abort`, la valeur est vide.
service_class_name	character(64)	Nom de la classe de service.

Exemples de requêtes

L'exemple suivant recherche les requêtes qui ont été arrêtées par une règle de surveillance de requête.

```
Select query, rule
from stl_wlm_rule_action
where action = 'abort'
order by query;
```

STL_WLM_QUERY

Contient un enregistrement de chaque tentative d'exécution d'une requête dans une classe de service gérée par WLM.

STL_WLM_QUERY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
xid	entier	ID de transaction de la requête ou sous-requête.

Nom de la colonne	Type de données	Description
tâche	entier	ID utilisé pour suivre une requête via le gestionnaire de la charge de travail. Peut être associé à plusieurs requêtes IDs. Si une requête est redémarrée, la requête se voit attribuer un nouvel ID de requête, mais pas un nouvel ID de tâche.
query	entier	ID de requête. Si une requête est redémarrée, la requête se voit attribuer un nouvel ID de requête, mais pas un nouvel ID de tâche.
service_class	entier	ID de la classe de service. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.
slot_count	entier	Nombre d'emplacements de requête WLM qu'une requête utilise conformément au niveau de concurrence défini pour la file d'attente. La valeur par défaut est 1. Pour plus d'informations, consultez wlm_query_slot_count.
service_class_start_time	timestamp	Heure à laquelle la requête a été attribuée à la classe de service. Cette heure est définie au fuseau horaire UTC.
queue_start_time	timestamp	Heure à laquelle la requête a été entrée dans la file d'attente pour la classe de service. Cette heure est définie au fuseau horaire UTC.
queue_end_time	timestamp	Heure à laquelle la requête a quitté la file d'attente pour la classe de service. Cette heure est définie au fuseau horaire UTC.
total_queue_time	bigint	Nombre total de microsecondes pendant lesquelles la requête se trouvait dans la file d'attente.

Nom de la colonne	Type de données	Description
exec_start_time	timestamp	Heure à laquelle la requête a commencé l'exécution dans la classe de service. Cette heure est définie au fuseau horaire UTC.
exec_end_time	timestamp	Heure à laquelle la requête a terminé l'exécution dans la classe de service. Cette heure est définie au fuseau horaire UTC.
total_exec_time	bigint	Nombre de microsecondes consacrées par la requête à l'exécution.
service_class_end_time	timestamp	Heure à laquelle la requête a quitté la classe de service. Cette heure est définie au fuseau horaire UTC.
final_state	character(16)	Réservé au système.
est_peak_mem	bigint	Réservé au système.
query_priority	char(20)	Priorité de la requête. Les valeurs possibles sont `n/a`, `lowest`, `low`, `normal`, `high` et `highest`, où `n/a` signifie que cette priorité de requête n'est pas prise en charge.
service_class_name	character(64)	Nom de la classe de service. Pour en savoir plus sur les classes de service, consultez Tables et vues système WLM.

Exemples de requêtes

Afficher la durée moyenne des requêtes dans les files d'attente et à l'exécution

Les requêtes suivantes affichent la configuration actuelle des classes de service supérieures à 4. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.

La requête suivante renvoie la durée moyenne (en microsecondes) que chaque requête a passé dans les files d'attente de requête et à l'exécution de chaque classe de service.

```
select service_class as svc_class, count(*),
avg(datediff(microseconds, queue_start_time, queue_end_time)) as avg_queue_time,
avg(datediff(microseconds, exec_start_time, exec_end_time )) as avg_exec_time
from stl_wlm_query
where service_class > 4
group by service_class
order by service_class;
```

Cette requête renvoie l'exemple de sortie suivant :

```
 svc_class | count | avg_queue_time | avg_exec_time
-----------+-------+----------------+---------------
         5 | 20103 |              0 |         80415
         5 |  3421 |          34015 |        234015
         6 |    42 |              0 |        944266
         7 |   196 |           6439 |       1364399
(4 rows)
```

Afficher la durée maximale des requêtes dans les files d'attente et à l'exécution

La requête suivante renvoie la durée maximale (en microsecondes) qu'une requête a passé dans une file d'attente de requêtes et à l'exécution de chaque classe de service.

```
select service_class as svc_class, count(*),
max(datediff(microseconds, queue_start_time, queue_end_time)) as max_queue_time,
max(datediff(microseconds, exec_start_time, exec_end_time )) as max_exec_time
from stl_wlm_query
where svc_class > 5
group by service_class
order by service_class;
```

```
 svc_class | count | max_queue_time | max_exec_time
-----------+-------+----------------+---------------
         6 |    42 |              0 |       3775896
         7 |   197 |          37947 |      16379473
(4 rows)
```

Tables STV pour les données d'instantanés

Les tables STV sont des tables système virtuelles contenant les instantanés des données système actuelles.

Rubriques

- STV_ACTIVE_CURSORS
- STV_BLOCKLIST
- STV_CURSOR_CONFIGURATION
- STV_DB_ISOLATION_LEVEL
- STV_EXEC_STATE
- STV_INFLIGHT
- STV_LOAD_STATE
- STV_LOCKS
- STV_ML_MODEL_INFO
- STV_MV_DEPS
- STV_MV_INFO
- STV_NODE_STORAGE_CAPACITY
- STV_PARTITIONS
- STV_QUERY_METRICS
- STV_RECENTS
- STV_SESSIONS
- STV_SLICES
- STV_STARTUP_RECOVERY_STATE
- STV_TBL_PERM
- STV_TBL_TRANS
- STV_WLM_CLASSIFICATION_CONFIG
- STV_WLM_QMR_CONFIG
- STV_WLM_QUERY_QUEUE_STATE
- STV_WLM_QUERY_STATE
- STV_WLM_QUERY_TASK_STATE
- STV_WLM_SERVICE_CLASS_CONFIG
- STV_WLM_SERVICE_CLASS_STATE
- STV_XRESTORE_ALTER_QUEUE_STATE

STV_ACTIVE_CURSORS

STV_ACTIVE_CURSORS affiche les détails des curseurs ouverts actuellement. Pour de plus amples informations, veuillez consulter DECLARE.

STV_ACTIVE_CURSORS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système. Un utilisateur ne peut voir que les curseurs ouverts par lui-même. Un super-utilisateur peut afficher tous les curseurs.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
name	characte (256)	Nom du curseur.
xid	bigint	Contexte de la transaction.
pid	entier	Processus principal exécutant la requête.
starttime	timestam	Heure à laquelle le curseur a été déclaré.
row_count	bigint	Nombre de lignes dans le jeu de résultats du curseur.
byte_count	bigint	Nombre d'octets dans le jeu de résultats du curseur.
fetched_rows	bigint	Nombre de lignes actuellement extraites du jeu de résultats du curseur.

STV_BLOCKLIST

STV_BLOCKLIST contient le nombre de blocs de disque de 1 Mo utilisés par chaque tranche, table ou colonne dans une base de données.

Pour déterminer le nombre de blocs de disque de 1 Mo alloués par base de données, table, tranche ou colonne, utilisez des requêtes agrégées avec STV_BLOCKLIST, comme dans les exemples suivants. Vous pouvez également utiliser STV_PARTITIONS afin d'afficher des informations de synthèse sur l'utilisation du disque.

STV_BLOCKLIST n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> STV_BLOCKLIST enregistre uniquement les blocs appartenant au cluster provisionné ou à l'espace de noms sans serveur. Si la base de données inclut des blocs partagés par un producteur de partage de données, ces blocs ne sont pas inclus dans STV_BLOCKLIST. Pour plus d'informations sur les partages de données, rendez-vous sur. Partage de données dans Amazon Redshift

Colonnes de la table

Nom de la colonne	Type de données	Description
slice	entier	Tranche de nœud.
col	entier	Index de base zéro de la colonne. Chaque table que vous créez possède trois colonnes masquées qui lui sont ajoutées : INSERT_XID, DELETE_XID et ROW_ID (OID). Une table avec 3 colonnes définies par l'utilisateur contient 6 colonnes réelles et les colonnes définies par l'utilisateur sont numérotées en interne 0, 1 et 2. Les colonnes INSERT_XID, DELETE_XID et ROW_ID sont numérotés respectivement 3, 4 et 5 dans cet exemple.
tbl	entier	ID de la table de la base de données.
blocknum	entier	ID du bloc de données.
num_values	entier	Nombre de valeurs contenues dans le bloc.

Nom de la colonne	Type de données	Description
extended_limits	entier	Pour utilisation interne.
minvalue	bigint	Valeur de données minimale du bloc. Stocke les huit premiers caractères en tant que nombre entier 64 bits pour des données non numériques. Utilisée pour l'analyse de disque.
maxvalue	bigint	Valeur de données maximale du bloc. Stocke les huit premiers caractères en tant que nombre entier 64 bits pour des données non numériques. Utilisée pour l'analyse de disque.
sb_pos	entier	Identifiant interne d'Amazon Redshift pour la position du super bloc sur le disque.
pinned	entier	Indique si le bloc est mis en mémoire dans le cadre du préchargement ou non. 0 = faux ; 1 = vrai. La valeur par défaut est false.
on_disk	entier	Indique si le bloc est automatiquement stocké sur le disque ou non. 0 = faux ; 1 = vrai. La valeur par défaut est false.
modifié	entier	Indique si le bloc a été modifié ou non. 0 = faux ; 1 = vrai. La valeur par défaut est false.
hdr_modified	entier	Indique si l'en-tête de bloc a été modifié ou non. 0 = faux ; 1 = vrai. La valeur par défaut est false.
unsorted	entier	Indique si un bloc est trié ou non. 0 = faux ; 1 = vrai. La valeur par défaut est true.
tombstone	entier	Pour utilisation interne.
preferred_diskno	entier	Numéro de disque sur lequel le bloc doit être, sauf si le disque a échoué. Une fois que le disque a été corrigé, le bloc renvoie sur ce disque.

Nom de la colonne	Type de données	Description
temporary	entier	Indique si le bloc contient ou non des données temporaires, comme une table temporaire ou les résultats intermédiaires des requêtes. 0 = faux ; 1 = vrai. La valeur par défaut est false.
newblock	entier	Indique si un bloc est nouveau (true) ou n'a jamais été validé sur le disque (false). 0 = faux ; 1 = vrai.
num_readers	entier	Nombre de références sur chaque bloc.
flags	entier	Indicateurs internes d'Amazon Redshift pour l'en-tête du bloc.

Exemples de requêtes

STV_BLOCKLIST contient une ligne par bloc de disque alloué, par conséquent une requête qui sélectionne toutes les lignes renvoie éventuellement un très grand nombre de lignes. Nous vous conseillons d'utiliser uniquement des requêtes agrégées avec STV_BLOCKLIST.

La vue SVV_DISKUSAGE fournit des informations similaires dans un format plus convivial. Toutefois, l'exemple suivant illustre une utilisation de la table STV_BLOCKLIST.

Pour déterminer le nombre de blocs de 1 Mo utilisés par chaque colonne dans la table VENUE, tapez la requête suivante :

```
select col, count(*)
from stv_blocklist, stv_tbl_perm
where stv_blocklist.tbl = stv_tbl_perm.id
and stv_blocklist.slice = stv_tbl_perm.slice
and stv_tbl_perm.name = 'venue'
group by col
order by col;
```

Cette requête renvoie le nombre de blocs de 1 Mo alloué à chaque colonne dans la table VENUE, illustré par les exemples de données suivants :

```
 col | count
```

```
-----+-------
   0 |  4
   1 |  4
   2 |  4
   3 |  4
   4 |  4
   5 |  4
   7 |  4
   8 |  4
(8 rows)
```

La requête suivante affiche ou non des données de table réellement distribuées à toutes tranches :

```
select trim(name) as table, stv_blocklist.slice, stv_tbl_perm.rows
from stv_blocklist,stv_tbl_perm
where stv_blocklist.tbl=stv_tbl_perm.id
and stv_tbl_perm.slice=stv_blocklist.slice
and stv_blocklist.id > 10000 and name not like '%#m%'
and name not like 'systable%'
group by name, stv_blocklist.slice, stv_tbl_perm.rows
order by 3 desc;
```

Cette requête génère l'exemple de sortie suivant, illustrant la distribution homogène des données de la table avec la plupart des lignes :

```
table   | slice | rows
----------+-------+-------
listing  |    13 | 10527
listing  |    14 | 10526
listing  |     8 | 10526
listing  |     9 | 10526
listing  |     7 | 10525
listing  |     4 | 10525
listing  |    17 | 10525
listing  |    11 | 10525
listing  |     5 | 10525
listing  |    18 | 10525
listing  |    12 | 10525
listing  |     3 | 10525
listing  |    10 | 10525
listing  |     2 | 10524
listing  |    15 | 10524
```

```
listing  |    16 | 10524
listing  |     6 | 10524
listing  |    19 | 10524
listing  |     1 | 10523
listing  |     0 | 10521
...
(180 rows)
```

La requête suivante détermine si tous les blocs désactivés ont été dédiés au disque :

```
select slice, col, tbl, blocknum, newblock
from stv_blocklist
where  tombstone > 0;

slice | col |   tbl  | blocknum | newblock
-------+-----+--------+----------+----------
4     |  0  | 101285 |    0     |   1
4     |  2  | 101285 |    0     |   1
4     |  4  | 101285 |    1     |   1
5     |  2  | 101285 |    0     |   1
5     |  0  | 101285 |    0     |   1
5     |  1  | 101285 |    0     |   1
5     |  4  | 101285 |    1     |   1
...
(24 rows)
```

STV_CURSOR_CONFIGURATION

STV_CURSOR_CONFIGURATION affiche les contraintes de configuration du curseur. Pour de plus amples informations, veuillez consulter Contraintes de curseur.

STV_CURSOR_CONFIGURATION n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
current_cursor_count	entier	Nombre de curseurs actuellement ouverts.
max_diskspace_usable	entier	Quantité d'espace disponible pour les curseurs, en mégaoctets. Cette contrainte s'appuie sur la taille du jeu de résultats du curseur maximal pour le cluster.
current_diskspace_used	entier	Quantité d'espace disque actuellement utilisé par les curseurs, en mégaoctets.

STV_DB_ISOLATION_LEVEL

STV_DB_ISOLATION_LEVEL affiche le niveau d'isolation actuel des bases de données. Pour plus d'informations sur les niveaux d'isolation, consultez CREATE DATABASE.

STV_DB_ISOLATION_LEVEL est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
db_name	characte (128)	Nom de la base de données.
isolation_level	characte (20)	Niveau d'isolation de la base de données. Les valeurs possibles incluent `Serializable` et `Snapshot Isolation` .

STV_EXEC_STATE

Utilisez la table STV_EXEC_STATE pour rechercher des informations sur les requêtes et les étapes de requête qui sont en cours d'exécution sur les nœuds de calcul.

Ces informations sont généralement utilisées uniquement pour résoudre des problèmes techniques. Les vues SVV_QUERY_STATE et SVL_QUERY_SUMMARY tirent leurs informations de STV_EXEC_STATE.

STV_EXEC_STATE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. Permet de joindre d'autres tables système et vues.
slice	entier	Tranche du nœud où l'étape s'est terminée.
segment	entier	Segment de la requête qui s'est exécutée. Un segment de la requête est une série d'étapes.
étape	entier	Étape du segment de la requête qui s'est terminée. Une étape est la plus petite unité exécutée par une requête.
starttime	timestamp	Heure à laquelle l'étape s'est exécutée.
currenttime	timestamp	Heure actuelle.

Nom de la colonne	Type de données	Description
tasknum	entier	Processus de tâche de requête qui est affecté à l'exécuti on de l'étape.
rows	bigint	Nombre de lignes traitées.
octets	bigint	Nombre d'octets traités.
étiquette	char(256)	Étiquette de l'étape, qui se compose d'un nom d'étape de requête et, le cas échéant, d'un ID de table et d'un nom de table (par exemple, `scan tbl=100448 name =user`). Les tables à trois chiffres font IDs généralem ent référence à des scans de tables transitoires. Lorsque `tbl=0` s'affiche, cela fait généralement référence à une analyse d'une valeur constante.
is_diskba sed	char(1)	Si cette étape de la requête a été effectuée comme une opération sur disque : true (**`t`**) ou false (**`f`**). Seules certaines étapes, telles que le hachage, le tri et l'agrégat ion, peuvent accéder au disque. La plupart des types d'étapes sont toujours effectués en mémoire.
workmem	bigint	Nombre d'octets de mémoire de travail assignés à l'étape.
num_parts	entier	Nombre de partitions entre lesquelles une table de hachage est divisée pendant une étape de hachage. Un nombre positif dans cette colonne n'implique pas que l'étape de hachage ait été exécutée comme opération sur disque. Vérifiez la valeur de la colonne IS_DISKBASED pour voir si l'étape de hachage était basée sur le disque.
is_rrscan	char(1)	Si la valeur est définie sur true (**`t`**), indique qu'une analyse à plage restreinte a été utilisée sur l'étape. La valeur par défaut est false (**`f`**).

Nom de la colonne	Type de données	Description
is_delayed_scan	char(1)	Si la valeur est définie sur true (**t**), indique qu'une analyse retardée a été utilisée sur l'étape. La valeur par défaut est false (**f**).

Exemples de requêtes

Plutôt que d'interroger directement STV_EXEC_STATE, Amazon Redshift recommande d'interroger SVL_QUERY_SUMMARY ou SVV_QUERY_STATE pour obtenir les informations dans STV_EXEC_STATE dans un format plus convivial. Pour en savoir plus, consultez la documentation de SVL_QUERY_SUMMARY ou SVV_QUERY_STATE.

STV_INFLIGHT

Utilisez la table STV_INFLIGHT pour déterminer les requêtes en cours d'exécution sur le cluster. En phase de résolution des problèmes, cela s'avère utile pour vérifier le statut des requêtes de longue durée.

STV_INFLIGHT n'indique pas les requêtes de nœud principal uniquement. Pour de plus amples informations, veuillez consulter Fonctions exécutées uniquement sur le nœud principal. STV_INFLIGHT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Résolution des problèmes avec STV_INFLIGHT

Si vous utilisez STV_INFLIGHT pour résoudre des problèmes de performances pour une requête ou un ensemble de requêtes, notez les points suivants :

- Les transactions ouvertes de longue durée accroissent généralement la charge. Ces transactions ouvertes peuvent allonger le temps d'exécution des autres requêtes.

- Les tâches COPY et ETL de longue durée peuvent affecter les autres requêtes s'exécutant sur le cluster, si elles consomment beaucoup de ressources de calcul. Dans la plupart des cas, le fait de décaler ces tâches de longue durée sur des périodes de faible utilisation a pour effet d'accroître les performances des charges de travail analytiques ou de création de rapports.
- Certaines vues fournissent des informations associées à STV_INFLIGHT. Tel est notamment le cas de STL_QUERYTEXT, qui capture le texte de requête pour les commandes SQL, et de SVV_QUERY_INFLIGHT, qui joint STV_INFLIGHT à STL_QUERYTEXT. Vous pouvez également utiliser STV_RECENTS avec STV_INFLIGHT pour la résolution des problèmes. Par exemple, STV_RECENTS peut indiquer si des requêtes spécifiques sont à l'état En cours d'exécution ou Terminé. En combinant ces informations avec les résultats de STV_INFLIGHT, vous pouvez en savoir plus sur les propriétés d'une requête et sur son impact sur les ressources de calcul.

Vous pouvez également surveiller les requêtes en cours d'exécution à l'aide de la console Amazon Redshift.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
slice	entier	Tranche sur laquelle la requête s'exécute.
query	entier	ID de requête. Permet de joindre d'autres tables système et vues.
étiquette	caractère (320)	Le nom du fichier utilisé pour exécuter la requête ou une étiquette définie avec une commande SET QUERY_GROUP. Si la requête n'est pas basée sur un fichier ou si le paramètre QUERY_GROUP n'est pas défini, le champ est vide.
xid	bigint	ID de transaction.
pid	entier	ID du processus. Toutes les requêtes d'une séance étant exécutées dans le même processus, cette valeur reste

Nom de la colonne	Type de données	Description
		constante si vous exécutez une série de requêtes dans la même séance. Vous pouvez utiliser cette colonne pour la joindre à la table STL_ERROR.
starttime	timestamp	Heure de début de la requête.
text	character(100)	Texte de requête, tronqué à 100 caractères, si l'instruction dépasse cette limite.
suspended	entier	Indique si la requête est suspendue ou non. 0 = faux ; 1 = vrai.
insert_pristine	entier	Si les requêtes are/were able to run while the current query is/was d'écriture sont en cours d'exécution. 1 = aucune requête d'écriture autorisée. 0 = requêtes d'écriture autorisées. Cette colonne est destinée à être utilisée dans le débogage.
concurrency_scaling_status	entier	Indique si la requête a été exécutée sur le cluster principal ou sur un cluster de mise à l'échelle de simultanéité. Les valeurs possibles sont les suivantes : 0 - Exécutée sur le cluster principal 1 - Exécutée sur un cluster de mise à l'échelle de simultanéité

Exemples de requêtes

Pour afficher toutes les requêtes actives en cours d'exécution sur la base de données, entrez la requête suivante :

```
select * from stv_inflight;
```

L'exemple de sortie ci-dessous illustre les deux requêtes en cours d'exécution, y compris la requête STV_INFLIGHT elle-même et une requête qui a été exécutée à partir d'un script appelé :`avgwait.sql`

```
select slice, query, trim(label) querylabel, pid,
starttime, substring(text,1,20) querytext
from stv_inflight;

slice|query|querylabel | pid |        starttime         |      querytext
-----+-----+-----------+-----+--------------------------+--------------------
1011 |  21 |           | 646 |2012-01-26 13:23:15.645503|select slice, query,
1011 |  20 |avgwait.sql| 499 |2012-01-26 13:23:14.159912|select avg(datediff(
(2 rows)
```

La requête suivante sélectionne plusieurs colonnes, dont concurrency_scaling_status. Cette colonne indique si des requêtes sont envoyées au cluster de mise à l'échelle de la simultanéité. Si la valeur est 1 pour certains résultats, cela indique que des ressources de calcul avec mise à l'échelle de la simultanéité sont utilisées. Pour de plus amples informations, veuillez consulter Mise à l'échelle de la simultanéité.

```
select userid,
query,
pid,
starttime,
text,
suspended,
concurrency_scaling_status
 from STV_INFLIGHT;
```

L'exemple de sortie montre qu'une requête est envoyée au cluster de mise à l'échelle de la simultanéité.

```
 query  | pid     |        starttime           |   text                | suspended
 |  concurrency_scaling_status
--------+---------
+---------------------------|-----------------------|--------------|-----------------------
1234567 | 123456  | 2012-01-26 13:23:15.645503 | select userid, query...  0
    1
2345678 | 234567  | 2012-01-26 13:23:14.159912 | select avg(datediff(...  0
    0
(2 rows)
```

Pour obtenir des conseils supplémentaires sur la résolution des problèmes liés aux performances des requêtes, consultez Résolution des problèmes de requêtes.

STV_LOAD_STATE

Utilisez la table STV_LOAD_STATE pour rechercher des informations sur l'état actuel des instructions COPY en cours.

La commande COPY met à jour ce tableau après chaque million d'enregistrements chargés.

STV_LOAD_STATE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
séance	entier	PID de session du processus effectuant la charge.
query	entier	ID de requête. Permet de joindre d'autres tables système et vues.
slice	entier	Numéro de la tranche de nœud.
pid	entier	ID du processus. Toutes les requêtes d'une séance étant exécutées dans le même processus, cette valeur reste constante si vous exécutez une série de requêtes dans la même séance.
recordtime	timestamp	Heure de l'enregistrement.
bytes_to_load	bigint	Nombre total d'octets à charger par cette tranche. Ce nombre est égal à 0 si les données en cours de chargement sont compressées
bytes_loaded	bigint	Nombre d'octets chargés par cette tranche. Si les données en cours de chargement sont compressées, il s'agit du

Nom de la colonne	Type de données	Description
		nombre d'octets chargés une fois que les données sont décompressées.
bytes_to_load_compressed	bigint	Nombre total d'octets de données compressées à charger par cette tranche. Ce nombre est égal à 0 si les données en cours de chargement ne sont pas compressées.
bytes_loaded_compressed	bigint	Nombre d'octets de données compressées chargés par cette tranche. Ce nombre est égal à 0 si les données en cours de chargement ne sont pas compressées.
lines	entier	Nombre de lignes chargées par cette tranche.
num_files	entier	Nombre de fichiers à charger par cette tranche.
num_files_complete	entier	Nombre de fichiers chargés par cette tranche.
current_file	character (256)	Nom du fichier en cours de chargement par cette tranche.
pct_complete	entier	Pourcentage de chargement de données réalisé par cette tranche.

Exemple de requête

Pour afficher la progression de chaque tranche pour une commande COPY, tapez la requête suivante. Cet exemple utilise la fonction PG_LAST_COPY_ID() pour récupérer des informations sur la dernière commande COPY.

```
select slice , bytes_loaded, bytes_to_load , pct_complete from stv_load_state where
 query = pg_last_copy_id();

 slice | bytes_loaded | bytes_to_load | pct_complete
-------+--------------+---------------+--------------
     2 |            0 |             0 |            0
     3 |     12840898 |      39104640 |           32
(2 rows)
```

STV_LOCKS

La table STV_LOCKS permet d'afficher les mises à jour actuelles dans les tables de la base de données.

Amazon Redshift verrouille les tables pour empêcher deux utilisateurs de mettre à jour la même table simultanément. Bien que le tableau STV_LOCKS affiche toutes les mises à jour de table actuelles, interrogez la table STL_TR_CONFLICT pour afficher un journal des conflits de verrouillage. Utilisez la vue SVV_TRANSACTIONS pour identifier les transactions en cours et les problèmes de conflit de verrous.

STV_LOCKS n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
table_id	bigint	ID de la table obtenant le verrou.
last_commit	timestamp	Horodatage de la dernière validation de la table.
last_update	timestamp	Horodatage de la dernière mise à jour de la table.
lock_owner	bigint	ID de transaction associé au verrou.
lock_owner_pid	bigint	ID de processus associé au verrou.
lock_owner_start_ts	timestamp	Horodatage de l'heure de début de la transaction.
lock_owner_end_ts	timestamp	Horodatage de l'heure de fin de la transaction.
lock_status	character (22)	État du processus en attente ou détenant un verrou.

Exemple de requête

Pour afficher tous les verrous appliqués aux transactions en cours, tapez la commande suivante :

```
select table_id, last_update, lock_owner, lock_owner_pid from stv_locks;
```

Cette requête renvoie l'exemple de sortie suivant, qui affiche trois verrous actuellement en vigueur :

```
 table_id |        last_update         | lock_owner | lock_owner_pid
----------+----------------------------+------------+----------------
100004  | 2008-12-23 10:08:48.882319 |       1043 |           5656
100003  | 2008-12-23 10:08:48.779543 |       1043 |           5656
100140  | 2008-12-23 10:08:48.021576 |       1043 |           5656
(3 rows)
```

STV_ML_MODEL_INFO

Informations sur l'état actuel du modèle de machine learning.

STV_ML_MODEL_INFO est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
nom_schéma	char(128)	L'espace de nom du modèle.
user_name	char(128)	Le propriétaire du modèle.
model_name	char(128)	Nom du modèle.
life_cycle	char(20)	L'état du cycle de vie du modèle.
is_refreshable	entier	L'état du modèle, à savoir s'il peut être actualisé si les tables et les colonnes originales de la requête d'entraînement existent toujours et si l'utilisateur a toujours les permissions pour les

Nom de la colonne	Type de données	Description
		utiliser. Les valeurs possibles sont : 1 (actualisable) et 0 (non actualisable).
model_state	char(128)	L'état actuel du modèle.

Exemple de requête

La requête suivante affiche l'état actuel des modèles de machine learning.

```
SELECT schema_name, model_name, model_state
FROM stv_ml_model_info;

 schema_name |        model_name            |             model_state
-------------+------------------------------+-------------------------------------
 public      | customer_churn_auto_model    | Train Model On SageMaker In Progress
 public      | customer_churn_xgboost_model | Model is Ready
(2 row)
```

STV_MV_DEPS

La table STV_MV_DEPS affiche les dépendances des vues matérialisées sur d'autres vues matérialisées dans Amazon Redshift.

Pour plus d'informations sur les vues matérialisées, consultez Vues matérialisées dans Amazon Redshift.

STV_MV_DEPS est visible par tous les utilisateurs. Les superutilisateurs peuvent voir toutes les lignes ; les utilisateurs ordinaires peuvent uniquement répertorier les vues matérialisées résidant dans les schémas auxquels ils ont accès. Pour plus d'informations, voir Visibilité des données dans les tables et les vues du système.

Colonnes de la table

Nom de la colonne	Type de données	Description
db_name	char(128)	La base de données qui contient la vue matérialisée spécifiée.
schema	char(128)	Schéma de la vue matérialisée.
name	char(128)	Nom de la vue matérialisée.
ref_schema	char(128)	Le schéma de vue matérialisée dont dépend cette vue matérialisée.
ref_name	char(128)	Le nom de la vue matérialisée dont dépend cette vue matérialisée.
ref_database_name	char(128)	Le nom de la base de données dont dépend cette vue matérialisée.

Exemple de requête

La requête suivante renvoie une ligne de sortie qui indique que la vue matérialisée `mv_over_foo` utilise la vue matérialisée `mv_foo` dans sa définition en tant que dépendance.

```
CREATE SCHEMA test_ivm_setup;
CREATE TABLE test_ivm_setup.foo(a INT);
CREATE MATERIALIZED VIEW test_ivm_setup.mv_foo AS SELECT * FROM test_ivm_setup.foo;
CREATE MATERIALIZED VIEW test_ivm_setup.mv_over_foo AS SELECT * FROM
 test_ivm_setup.mv_foo;

SELECT * FROM stv_mv_deps;

 db_name | schema          | name        |   ref_schema   | ref_name |
 ref_database_name
---------+-----------------+-------------+----------------+----------
+------------------
 dev     | test_ivm_setup  | mv_over_foo | test_ivm_setup | mv_foo   | dev
```

STV_MV_INFO

La table STV_MV_INFO contient une ligne pour chaque vue matérialisée, si les données sont obsolètes, ainsi que des informations d'état.

Pour plus d'informations sur les vues matérialisées, consultez Vues matérialisées dans Amazon Redshift.

STV_MV_INFO est visible par tous les utilisateurs. Les superutilisateurs peuvent voir toutes les lignes ; les utilisateurs ordinaires peuvent uniquement répertorier les vues matérialisées résidant dans les schémas auxquels ils ont accès. Pour plus d'informations, voir Visibilité des données dans les tables et les vues du système.

Colonnes de la table

Nom de la colonne	Type de données	Description
db_name	char(128)	Base de données contenant la vue matérialisée.
schema	char(128)	Schéma de la base de données.
name	char(128)	Nom de la vue matérialisée.
updated_upto_xid	bigint	Réservé pour un usage interne.
is_stale	char(1)	`t` indique que la vue matérialisée est obsolète. Une vue matérialisée obsolète est une vue dans laquelle les tables de base ont été mises à jour mais où la vue matérialisée n'a pas été actualisée. Les informations peuvent ne pas être exactes si aucune actualisation n'a été exécutée depuis le dernier redémarrage. La colonne `is_stale` est toujours définie sur `t` si la vue matérialisée dépend d'une fonction mutable. Une fonction mutable renvoie un résultat différent si le ou les arguments qui lui sont fournis

Nom de la colonne	Type de données	Description
		sont identiques. Par exemple, la plupart des fonctions qui renvoient une date ou un horodatage sont des fonctions mutables.
owner_user_name	char(128)	Utilisateur propriétaire de la vue matérialisée.
state	entier	État de la vue matérialisée comme suit : • 0 – La vue matérialisée est entièrement recalculée lors de l'actualisation. • 1 – La vue matérialisée est incrémentielle. • 101 – La vue matérialisée ne peut pas être actualisée en raison d'une colonne supprimée. Cette contrainte s'applique même si la colonne n'est pas utilisée dans la vue matérialisée. • 102 – La vue matérialisée ne peut pas être actualisée en raison d'un type de colonne modifié. Cette contrainte s'applique même si la colonne n'est pas utilisée dans la vue matérialisée. • 103 – La vue matérialisée ne peut pas être actualisée en raison d'une table renommée. • 104 – La vue matérialisée ne peut pas être actualisée en raison d'une colonne renommée. Cette contrainte s'applique même si la colonne n'est pas utilisée dans la vue matérialisée. • 105 – La vue matérialisée ne peut pas être actualisée en raison d'un schéma renommé.
réécriture automatique	char(1)	Un `t` indique que la vue matérialisée est éligible pour la réécriture automatique des requêtes.

Nom de la colonne	Type de données	Description
actualisation automatique	char(1)	Un `t` indique que la vue matérialisée peut être automatiquement actualisée.

Exemple de requête

Pour afficher l'état de toutes les vues matérialisées, exécutez la requête suivante.

```
select * from stv_mv_info;
```

Cette requête renvoie l'exemple de sortie suivant.

```
 db_name |       schema       |   name   | updated_upto_xid | is_stale | owner_user_name
 | state | autorefresh | autorewrite
---------+--------------------+---------+------------------+----------
+-----------------+-------+-------------+------------
 dev     | test_ivm_setup     | mv       |             1031 | f        | catch-22
 |     1 |           1 |           0
 dev     | test_ivm_setup     | old_mv   |              988 | t        | lotr
 |     1 |           0 |           1
```

STV_NODE_STORAGE_CAPACITY

La table STV_NODE_STORAGE_CAPACITY affiche les détails de la capacité de stockage totale et de la capacité totale utilisée pour chaque nœud d'un cluster. Il contient une ligne pour chaque nœud.

STV_NODE_STORAGE_CAPACITY n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
node	entier	Numéro du nœud.

Nom de la colonne	Type de données	Description
used	entier	Le nombre de blocs de disque de 1 Mo actuellement utilisés sur le nœud. Pour les types de RA3 nœuds, les blocs utilisés incluent à la fois les blocs mis en cache localement et les blocs conservés dans Amazon S3.
capacity	entier	La capacité de stockage totale allouée pour le nœud en blocs de 1 Mo. La capacité inclut l'espace réservé par Amazon Redshift sur les types de DC2 nœuds pour un usage interne. La capacité est plus grande que la capacité nominale du nœud, qui est la quantité d'espace du nœud disponible pour les données de l'utilisateur. Pour les types de RA3 nœuds, cette capacité est identique au quota de stockage géré total pour le cluster. Pour plus d'informations sur la capacité par type de nœud, consultez Détails de type de nœud dans le Guide de gestion Amazon Redshift.

Exemples de requêtes

> **Note**
>
> Les résultats des exemples suivants varient en fonction des spécifications des nœuds de votre cluster. Ajouter la colonne `capacity` à votre SQL SELECT pour récupérer la capacité de votre cluster.

La requête suivante renvoie l'espace utilisé et la capacité totale en blocs de disque de 1 Mo. Cet exemple a été exécuté sur un cluster à deux nœuds dc2.8xlarge.

```
select node, used from stv_node_storage_capacity order by node;
```

Cette requête renvoie l'exemple de sortie suivant.

```
 node | used
------+-------
    0 | 30597
    1 | 27089
```

La requête suivante renvoie l'espace utilisé et la capacité totale en blocs de disque de 1 Mo. Cet exemple a été exécuté sur un cluster à deux nœuds ra3.16xlarge.

```
select node, used from stv_node_storage_capacity order by node;
```

Cette requête renvoie l'exemple de sortie suivant.

```
 node | used
------+-------
    0 | 30591
    1 | 27103
```

STV_PARTITIONS

Utilisez la table STV_PARTITIONS pour connaître les performances de vitesse et d'utilisation des disques pour Amazon Redshift.

STV_PARTITIONS contient une ligne par nœud, par volume de disque logique.

STV_PARTITIONS n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
owner	entier	Nœud de disque qui possède la partition.
hôte	entier	Nœud qui est physiquement attaché à la partition.
diskno	entier	Disque contenant la partition.

Nom de la colonne	Type de données	Description
part_begin	bigint	Décalage de la partition. Les périphériques bruts sont partition nés logiquement pour ouvrir un espace pour les blocs miroirs.
part_end	bigint	Fin de la partition.
used	entier	Nombre de blocs de disque de 1 Mo actuellement en cours d'utilisation sur la partition.
tossed	entier	Nombre de blocs prêts à être supprimés, mais qui ne sont pas encore supprimés, car la libération de leurs adresses de disque n'est pas sécurisée. Si les adresses ont été libérées immédiatement, une transaction en attente peut écrire au même emplacement sur le disque. Par conséquent, ces blocs rejetés sont libérés à la prochaine validation. Les blocs de disque peuvent être marqués comme rejetés, par exemple, lorsqu'une colonne de table est supprimée, au cours des opérations INSERT, ou au cours des opérations de requête basées sur disque.
capacity	entier	Capacité totale de la partition dans les blocs de disque de 1 Mo.
reads	bigint	Nombre de lectures réalisées depuis le dernier redémarrage du cluster.
writes	bigint	Nombre d'écritures réalisées depuis le dernier redémarrage du cluster.
seek_forward	entier	Nombre de fois qu'une demande ne concerne pas l'adresse suivante, étant donné l'adresse de demande précédente.
seek_back	entier	Nombre de fois qu'une demande ne concerne pas l'adresse précédente, étant donné l'adresse de demande suivante.
is_san	entier	Si la partition appartient à un SAN. Les valeurs valides sont **0** (false) ou **1** (true).

Nom de la colonne	Type de données	Description
failed	entier	Cette colonne est obsolète.
mbps	entier	Vitesse du disque en mégabits par seconde.
mount	character (256)	Chemin du répertoire à l'appareil.

Exemple de requête

La requête suivante renvoie l'espace et la capacité utilisés sur le disque, dans des blocs de disque de 1 Mo et calcule l'utilisation du disque sous forme de pourcentage de l'espace sur le disque brut. L'espace disque brut comprend l'espace qui est réservé par Amazon Redshift pour un usage interne, il est donc plus grand que la capacité nominale du disque, qui est la quantité d'espace disque disponible pour l'utilisateur. La métrique Pourcentage d'espace disque utilisé sur l'onglet Performances de la console de gestion Amazon Redshift indique le pourcentage de capacité de disque nominale utilisée par votre cluster. Nous vous recommandons de surveiller la métrique Pourcentage d'espace disque utilisé pour maintenir votre utilisation dans la capacité de disque nominale de votre cluster.

> **Important**
>
> Nous vous recommandons vivement de ne pas dépasser la capacité de disque nominale de votre cluster. Bien que cela soit techniquement possible dans certains cas, dépasser la capacité nominale de votre disque diminue la tolérance aux pannes de votre cluster et augmente le risque de perte de données.

Cet exemple a été exécuté sur un cluster à deux nœuds avec six partitions de disque logique par nœud. L'espace est utilisé de manière très uniforme entre les disques, avec environ 25 % de chaque disque en cours d'utilisation.

```
select owner, host, diskno, used, capacity,
(used-tossed)/capacity::numeric *100 as pctused
from stv_partitions order by owner;
```

```
 owner | host | diskno |  used  | capacity | pctused
-------+------+--------+--------+----------+---------
   0   |  0   |    0   | 236480 |  949954  | 24.9
   0   |  0   |    1   | 236420 |  949954  | 24.9
   0   |  0   |    2   | 236440 |  949954  | 24.9
   0   |  1   |    2   | 235150 |  949954  | 24.8
   0   |  1   |    1   | 237100 |  949954  | 25.0
   0   |  1   |    0   | 237090 |  949954  | 25.0
   1   |  1   |    0   | 236310 |  949954  | 24.9
   1   |  1   |    1   | 236300 |  949954  | 24.9
   1   |  1   |    2   | 236320 |  949954  | 24.9
   1   |  0   |    2   | 237910 |  949954  | 25.0
   1   |  0   |    1   | 235640 |  949954  | 24.8
   1   |  0   |    0   | 235380 |  949954  | 24.8

(12 rows)
```

STV_QUERY_METRICS

Contient des informations sur les métriques, telles que le nombre total de lignes traitées et d'opérations d'entrée/sortie, l'utilisation de l'UC et l'utilisation du disque pour les requêtes actives s'exécutant dans les files d'attente de requêtes définies par l'utilisateur (classes de service). Pour afficher les métriques des requêtes qui ont été terminées, consultez la table système STL_QUERY_METRICS.

Les métriques de requête sont échantillonnées toutes les secondes. Par conséquent, la même requête exécutée plusieurs fois peut renvoyer des heures légèrement différentes. En outre, il est possible que les segments de requête qui s'exécutent en moins d'une seconde ne soient pas enregistrés.

STV_QUERY_METRICS suit et regroupe les métriques au niveau de la requête, du segment et de l'étape. Pour plus d'informations sur les segments et les étapes de requête, consultez Workflow d'exécution et de planification de requête. De nombreuses métriques (`max_rows`, `cpu_time`, etc.) sont additionnées entre les tranches de nœuds. Pour plus d'informations sur les tranches de nœuds, consultez Architecture système de l'entrepôt des données.

Pour déterminer le niveau auquel la ligne communique les métriques, observez les colonnes `segment` et `step_type` :

- Si `segment` et `step_type` sont définis sur `-1`, la ligne communique les métriques au niveau de la requête.

- Si `segment` n'est pas défini sur `-1` et si `step_type` est défini sur `-1`, la ligne communique les métriques au niveau du segment.
- Si `segment` et `step_type` ne sont pas définis sur `-1`, la ligne communique les métriques au niveau de l'étape.

STV_QUERY_METRICS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a exécuté la requête qui a généré l'entrée.
service_class	entier	ID de la file d'attente de requêtes WLM (classe de service). Les files d'attente de requêtes sont définies dans la configura tion WLM. Les métriques sont communiquées uniquement pour les files d'attente définies par l'utilisateur.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
starttime	timestamp	Heure au format UTC de début de l'exécution de la requête, avec 6 chiffres de précision pour les fractions de secondes. Par exemple : `2009-06-12 11:29:19.131358` .
slices	entier	Nombre de tranches du cluster.
segment	entier	Numéro de segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes.

Nom de la colonne	Type de données	Description
		Les segments de requête peuvent s'exécuter en parallèle . Chaque segment s'exécute dans un processus unique. Si le segment a une valeur de -1, les valeurs du segment de métriques sont reportées au niveau de la requête.
step_type	entier	Type de l'étape exécutée. Pour obtenir une description des types d'étapes, consultez Types d'étapes.
rows	bigint	Nombre de lignes traitées par une étape.
max_rows	bigint	Nombre maximal de lignes produites pour une étape, regroupées entre toutes les tranches.
cpu_time	bigint	Durée d'utilisation de l'UC, en microsecondes. Au niveau du segment, durée totale d'utilisation de l'UC pour le segment entre toutes les tranches. Au niveau de la requête, somme des durées d'utilisation de l'UC pour la requête entre toutes les tranches et tous les segments.
max_cpu_time	bigint	Durée maximale d'utilisation de l'UC, en microsecondes. Au niveau du segment, durée maximale d'utilisation de l'UC par le segment entre toutes les tranches. Au niveau de la requête, durée maximale d'utilisation de l'UC par n'importe quel segment de requête.
blocks_read	bigint	Nombre de blocs d'1 Mo lus par la requête ou le segment.
max_block s_read	bigint	Nombre maximal de blocs d'1 Mo lus par le segment, regroupés entre toutes les tranches. Au niveau du segment, nombre maximal de blocs d'1 Mo lus pour le segment entre toutes les tranches. Au niveau de la requête, nombre maximal de blocs d'1 Mo lus par n'importe quel segment de la requête.

Nom de la colonne	Type de données	Description
run_time	bigint	Somme des durées totales d'exécution, entre les tranches. La durée d'attente n'est pas incluse dans le délai d'exécution. Au niveau du segment, somme des durées d'exécution pour le segment entre toutes les tranches. Au niveau de la requête, somme des durées d'exécution pour la requête entre toutes les tranches et tous les segments. Puisqu'il s'agit d'une somme, le délai d'exécution n'est pas lié au délai d'exécution de la requête.
max_run_time	bigint	Durée maximale écoulée pour un segment (en microseco ndes). Au niveau du segment, durée maximale d'exécution pour le segment entre toutes les tranches. Au niveau de la requête, durée maximale d'exécution pour n'importe quel segment de requête.
max_block s_to_disk	bigint	Quantité maximale d'espace disque utilisé pour écrire des résultats intermédiaires, en blocs de 1 Mo. Au niveau du segment, quantité maximale d'espace disque utilisé par le segment entre toutes les tranches. Au niveau de la requête, quantité maximale d'espace disque utilisé par n'importe quel segment de requête.
blocks_to_disk	bigint	Quantité d'espace disque utilisé par une requête ou un segment pour écrire des résultats intermédiaires, en blocs de 1 Mo.
étape	entier	Étape de la requête exécutée.
max_query _scan_size	bigint	Taille maximale des données analysées par une requête, en Mo. Au niveau du segment, taille maximale des données analysées par le segment entre toutes les tranches. Au niveau de la requête, taille maximale des données analysées par n'importe quel segment de requête.

Nom de la colonne	Type de données	Description
query_scan_size	bigint	Taille des données analysées par une requête, en Mo.
query_priority	entier	Priorité de la requête. Les valeurs possibles sont `-1`, `0`, `1`, `2`, `3` et `4`, où `-1` signifie que cette priorité de requête n'est pas prise en charge.
query_queue_time	bigint	Durée, en microsecondes, pendant laquelle la requête a été mise en file d'attente.

Types d'étapes

Le tableau suivant répertorie les types d'étapes relatifs aux utilisateurs des bases de données. Les types d'étapes relatifs à un usage interne uniquement ne sont pas inclus. Si le type d'étape est -1, la métrique n'est pas communiquée au niveau de l'étape.

Type d'étape	Description
1	Analyser la table
2	Insérer des lignes
3	Regrouper les lignes
6	Étape de tri
7	Étape de fusion
8	Étape de distribution
9	Étape de distribution de diffusion
10	Joindre par hachage
11	Joindre par fusion
12	Étape d'enregistrement

Type d'étape	Description
14	Hachage
15	Jointure de boucle imbriquée
16	Champs et expressions du projet
17	Limite le nombre de lignes retournées
18	Unique
20	Supprimer les lignes
26	Limite le nombre de lignes triées retournées
29	Calcule une fonction de fenêtre
32	UDF
33	Unique
37	Retourne au client les lignes issues du nœud principal
38	Retourne au nœud principal les lignes issues des nœuds de calcul
40	Analyse du spectre

Exemple de requête

Pour rechercher des requêtes actives avec un temps UC élevé (plus de 1 000 secondes), exécutez la requête suivante.

```
select query, cpu_time / 1000000 as cpu_seconds
from stv_query_metrics where segment = -1 and cpu_time > 1000000000
order by cpu_time;

query | cpu_seconds
------+------------
25775 |        9540
```

Pour rechercher des requêtes actives avec une jonction de boucles imbriquées qui a renvoyé plus d'un million de lignes, exécutez la requête suivante.

```
select query, rows
from stv_query_metrics
where step_type = 15 and rows > 1000000
order by rows;

query | rows
------+-----------
25775 | 1580225854
```

Pour rechercher des requêtes actives qui se sont exécutées pendant plus de 60 secondes et ont utilisé moins de 10 secondes de temps UC, exécutez la requête suivante.

```
select query, run_time/1000000 as run_time_seconds
from stv_query_metrics
where segment = -1 and run_time > 60000000 and cpu_time < 10000000;

query | run_time_seconds
------+-----------------
25775 |              114
```

STV_RECENTS

Utilisez la table STV_RECENTS pour trouver des informations sur les requêtes actuellement actives et récemment exécutées sur une base de données.

STV_RECENTS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Résolution des problèmes avec STV_RECENTS

STV_RECENTS est particulièrement utile pour déterminer si une requête ou un ensemble de requêtes sont en cours d'exécution ou terminées. Elle indique également la durée d'exécution d'une requête. Cela est utile pour identifier les requêtes qui prennent du temps à s'exécuter.

Vous pouvez joindre STV_RECENTS à d'autres vues système, telles que STV_INFLIGHT, de façon à recueillir des métadonnées supplémentaires sur les requêtes en cours d'exécution. (Vous trouverez dans la section des exemples de requêtes un exemple qui montre comment procéder.) Vous pouvez également utiliser les enregistrements renvoyés par cette vue ainsi que les fonctionnalités de surveillance de la console Amazon Redshift pour résoudre les problèmes en temps réel.

Parmi les vues système qui complètent STV_RECENTS figurent STL_QUERYTEXT, qui extrait le texte de requête des commandes SQL, et SVV_QUERY_INFLIGHT, qui joint STV_INFLIGHT à STL_QUERYTEXT.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
status	character(20)	Statut de la requête. Les valeurs valides sont **`Running`**, **`Done`**.
starttime	timestamp	Heure de début de la requête.
duration	entier	Nombre de microsecondes depuis que la session a commencé.
user_name	character(50)	Nom de l'utilisateur qui a lancé le processus.
db_name	character(50)	Nom du moteur de la base de données.
query	character(600)	Texte de la requête, jusqu'à 600 caractères. Tous les caractères supplémentaires sont tronqués.
pid	entier	ID du processus de la session associée à la requête, qui est toujours -1 pour les requêtes qui ont été terminées.

Exemples de requêtes

Pour identifier les requêtes qui sont en cours d'exécution sur la base de données, exécutez la requête suivante :

```
select user_name, db_name, pid, query
from stv_recents
where status = 'Running';
```

L'exemple de sortie ci-dessous illustre une seule requête en cours d'exécution sur la base de données TICKIT :

```
user_name | db_name |   pid   | query
----------+---------+---------+-------------
dwuser    | tickit  |  19996  |select venuename, venueseats from
venue where venueseats > 50000 order by venueseats desc;
```

L'exemple suivant renvoie la liste des requêtes (le cas échéant) qui sont en cours d'exécution ou qui se trouvent dans une file d'attente en attendant d'être exécutées :

```
select * from stv_recents where status<>'Done';

status |    starttime        | duration |user_name|db_name| query     | pid
-------+---------------------+----------+---------+-------+-----------+------
Running| 2010-04-21 16:11... | 281566454| dwuser  |tickit | select ...| 23347
```

Cette requête ne renvoie pas de résultats, sauf si vous exécutez un certain nombre de requêtes simultanées et que certaines d'entre elles se trouvent dans une file d'attente.

L'exemple suivant développe l'exemple précédent. Dans ce cas, les requêtes qui sont vraiment « en cours d'exécution » (pas en attente) sont exclues du résultat :

```
select * from stv_recents where status<>'Done'
and pid not in (select pid from stv_inflight);
...
```

Pour obtenir des conseils supplémentaires sur la résolution des problèmes liés aux performances des requêtes, consultez Résolution des problèmes de requêtes.

STV_SESSIONS

Utilisez la table STV_SESSIONS pour afficher des informations sur les sessions utilisateur actives pour Amazon Redshift.

Pour afficher l'historique des sessions, utilisez la table STL_SESSIONS, plutôt que STV_SESSIONS.

STV_SESSIONS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_SESSION_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
starttime	timestamp	Heure de démarrage de la session.
process	entier	ID de processus de la séance.
user_name	character(50)	Utilisateur associé à la session.
db_name	character(50)	Nom de la base de données associée à la séance.
timeout_sec	int	La durée maximale en secondes pendant laquelle une session reste inactive ou en veille avant de se terminer. 0 indique qu'aucun délai d'attente n'est défini.

Exemples de requêtes

Pour effectuer une vérification rapide afin de voir si d'autres utilisateurs sont actuellement connectés à Amazon Redshift, tapez la requête suivante :

```
select count(*)
```

```
from stv_sessions;
```

Si le résultat est supérieur à un, alors au moins un autre utilisateur est actuellement connecté à la base de données.

Pour afficher toutes les sessions actives pour Amazon Redshift, tapez la requête suivante :

```
select *
from stv_sessions;
```

Le résultat suivant montre quatre sessions actives fonctionnant sur Amazon Redshift :

```
       starttime         | process |user_name                   | db_name
      | timeout_sec
-------------------------+---------+----------------------------
+---------------------------+-------------
  2018-08-06 08:44:07.50 |   13779 | IAMA:aws_admin:admin_grp   | dev
      | 0
  2008-08-06 08:54:20.50 |   19829 | dwuser                     | dev
      | 120
  2008-08-06 08:56:34.50 |   20279 | dwuser                     | dev
      | 120
  2008-08-06 08:55:00.50 |   19996 | dwuser                     | tickit
      | 0
(3 rows)
```

Le nom d'utilisateur préfixé par IAMA indique que l'utilisateur s'est connecté à l'aide de l'authentification unique fédérée. Pour plus d'informations, consultez Utilisation de l'authentification IAM pour générer des informations d'identification de l'utilisateur de base de données.

STV_SLICES

Utilisez la table STV_SLICES pour afficher le mappage actuel d'une tranche à un nœud.

Les informations contenues dans STV_SLICES sont utilisées principalement à des fins d'enquête.

STV_SLICES est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
node	entier	Nœud du cluster où se trouve la tranche.
slice	entier	Tranche de nœud.
localslice	entier	Information à utilisation interne uniquement.
type	character (1)	Information à utilisation interne uniquement.

Exemple de requête

Pour afficher les nœuds du cluster qui gèrent les tranches, tapez la requête suivante :

```
select node, slice from stv_slices;
```

Cette requête renvoie l'exemple de sortie suivant :

```
 node | slice
------+-------
    0 |     2
    0 |     3
    0 |     1
    0 |     0
(4 rows)
```

STV_STARTUP_RECOVERY_STATE

Enregistre l'état des tables qui sont temporairement verrouillées pendant les opérations de redémarrage du cluster. Amazon Redshift place un verrou temporaire sur les tables pendant qu'elles sont traitées pour résoudre les transactions devenues obsolètes à la suite d'un redémarrage du cluster.

STV_STARTUP_RECOVERY_STATE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs

propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
db_id	entier	ID de base de données.
table_id	entier	ID de table.
table_name	character(137)	Nom de la table.

Exemples de requêtes

Pour contrôler quelles tables sont temporairement verrouillées, exécutez la requête suivante après le redémarrage d'un cluster.

```
select * from STV_STARTUP_RECOVERY_STATE;

  db_id | tbl_id | table_name
--------+--------+------------
 100044 | 100058 | lineorder
 100044 | 100068 | part
 100044 | 100072 | customer
 100044 | 100192 | supplier
(4 rows)
```

STV_TBL_PERM

La table STV_TBL_PERM contient des informations sur les tables permanentes dans Amazon Redshift, y compris les tables temporaires créées par un utilisateur pour la session en cours. STV_TBL_PERM contient des informations pour toutes les tables de toutes les bases de données.

Ce tableau diffère de STV_TBL_TRANS, qui contient des informations sur les tables de base de données temporaires que le système crée pendant le traitement de la requête.

STV_TBL_PERM n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
slice	entier	Tranche de nœud allouée à la table.
id	entier	ID de table.
name	character (72)	Nom de la table.
rows	bigint	Nombre de lignes de données dans la tranche.
sorted_rows	bigint	Nombre de lignes dans la tranche qui sont déjà triées sur le disque. Si ce nombre ne correspond pas au nombre ROWS, effectuez un processus vacuum sur la table pour retrier les lignes.
temp	entier	Si la table est une table temporaire ou non. 0 = false ; 1 = vrai.
db_id	entier	ID de la base de données dans laquelle la table a été créée.
insert_pristine	entier	Pour utilisation interne.
delete_pristine	entier	Pour utilisation interne.
backup	entier	Valeur qui indique si la table est incluse dans les instantanés de cluster. 0 = non ; 1 = oui. Pour plus d'informations, consultez le paramètre BACKUP de la commande CREATE TABLE.
dist_style	entier	Style de distribution de la table à laquelle appartient la tranche. Pour plus d'informations sur les valeurs, consultez Affichage des styles de distribution. Pour plus d'informations sur les styles de distribution, consultez Styles de distribution.

Nom de la colonne	Type de données	Description
block_count	entier	Nombre de blocs utilisés par la tranche. La valeur est -1 lorsque le nombre de blocs ne peut pas être calculé.

Exemples de requêtes

La requête suivante renvoie une liste de tables IDs et de noms distincts :

```
select distinct id, name
from stv_tbl_perm order by name;

   id   |          name
--------+------------------------
 100571 | category
 100575 | date
 100580 | event
 100596 | listing
 100003 | padb_config_harvest
 100612 | sales
...
```

D'autres tables système utilisent une table IDs. Il peut donc être très utile de savoir quel ID de table correspond à une table donnée. Dans cet exemple, SELECT DISTINCT est utilisé pour supprimer les doublons (les tables sont réparties entre plusieurs tranches).

Pour déterminer le nombre de blocs de 1 Mo utilisés par chaque colonne dans la table VENUE, tapez la requête suivante :

```
select col, count(*)
from stv_blocklist, stv_tbl_perm
where stv_blocklist.tbl = stv_tbl_perm.id
and stv_blocklist.slice = stv_tbl_perm.slice
and stv_tbl_perm.name = 'venue'
group by col
order by col;

 col | count
-----+-------
```

```
   0 |     8
   1 |     8
   2 |     8
   3 |     8
   4 |     8
   5 |     8
   6 |     8
   7 |     8
(8 rows)
```

Notes d'utilisation

La colonne ROWS inclut le nombre de lignes supprimées qui n'ont pas été vidées (ou ont été vidées mais avec l'option SORT ONLY). Par conséquent, la SUM de la colonne ROWS de la table STV_TBL_PERM peut ne pas correspondre au résultat COUNT(*) lorsque vous interrogez une table donnée directement. Par exemple, si 2 lignes sont supprimées de la table VENUE, le résultat de COUNT(*) est 200 mais le résultat de SUM(ROWS) est toujours 202 :

```
delete from venue
where venueid in (1,2);

select count(*) from venue;
count
-------
200
(1 row)

select trim(name) tablename, sum(rows)
from stv_tbl_perm where name='venue' group by name;

tablename | sum
-----------+-----
venue     | 202
(1 row)
```

Pour synchroniser les données dans STV_TBL_PERM, effectuez un processus vacuum sur la table VENUE.

```
vacuum venue;

select trim(name) tablename, sum(rows)
from stv_tbl_perm
```

```
where name='venue'
group by name;

tablename | sum
-----------+-----
venue     | 200
(1 row)
```

STV_TBL_TRANS

Utilisez la table STV_TBL_TRANS pour rechercher des informations sur les tables de base de données temporaires qui sont actuellement en mémoire.

Les tables temporaires sont des jeux de lignes généralement temporaires qui sont utilisés en tant que résultats intermédiaires pendant qu'une requête s'exécute. STV_TBL_TRANS diffère de STV_TBL_PERM du fait que STV_TBL_PERM contient des informations sur les tables de base de données permanentes.

STV_TBL_TRANS n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
slice	entier	Tranche de nœud allouée à la table.
id	entier	ID de table.
rows	bigint	Nombre de lignes de données dans la table.
size	bigint	Nombre d'octets alloué à la table.
query_id	bigint	ID de requête.
ref_cnt	entier	Nombre de références.
from_suspended	entier	Si cette table a été créée au cours d'une requête qui est désormais suspendue, ou non.

Nom de la colonne	Type de données	Description
prep_swap	entier	Si cette table temporaire est prête à passer au disque si nécessaire, ou non. (L'échange se produit uniquement dans les cas où la mémoire est faible.)

Exemples de requêtes

Pour afficher les informations de table temporaires pour une requête avec un ID de requête de 90, tapez la commande suivante :

```
select slice, id, rows, size, query_id, ref_cnt
from stv_tbl_trans
where query_id = 90;
```

Cette requête renvoie les informations de table temporaire pour la requête 90, comme illustré dans l'exemple de sortie suivant :

```
slice | id | rows | size | query_ | ref_ | from_     | prep_
      |    |      |      | id     | cnt  | suspended | swap
------+----+------+------+--------+------+-----------+-------
 1013 | 95 |    0 |    0 |     90 |    4 |         0 |   0
    7 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   10 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   17 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   14 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    3 | 96 |    0 |    0 |     90 |    4 |         0 |   0
 1013 | 99 |    0 |    0 |     90 |    4 |         0 |   0
    9 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    5 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   19 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    2 | 96 |    0 |    0 |     90 |    4 |         0 |   0
 1013 | 98 |    0 |    0 |     90 |    4 |         0 |   0
   13 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    1 | 96 |    0 |    0 |     90 |    4 |         0 |   0
 1013 | 96 |    0 |    0 |     90 |    4 |         0 |   0
    6 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   11 | 96 |    0 |    0 |     90 |    4 |         0 |   0
```

```
   15 | 96 |    0 |    0 |     90 |    4 |         0 |   0
   18 | 96 |    0 |    0 |     90 |    4 |         0 |   0
```

Dans cet exemple, vous pouvez voir que les données de la requête comportent des tables 95, 96 et 98. Étant donné que zéro octet est alloué à cette table, cette requête peut s'exécuter dans la mémoire.

STV_WLM_CLASSIFICATION_CONFIG

Contient les règles de classification actuelles pour WLM.

STV_WLM_CLASSIFICATION_CONFIG n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
id	entier	ID de classe de service. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.
condition	character(128)	Conditions de requête.
action_seq	entier	Réservé au système.
action	character(64)	Réservé au système.
action_service_class	entier	La classe de service dans laquelle l'action se déroule.

Exemple de requête

```
select * from STV_WLM_CLASSIFICATION_CONFIG;

id | condition                                    | action_seq | action |
 action_service_class
---+----------------------------------------------+------------+--------
+---------------------
```

```
 1 | (system user) and (query group: health)     |          0 | assign |
     1
 2 | (system user) and (query group: metrics)    |          0 | assign |
     2
 3 | (system user) and (query group: cmstats)    |          0 | assign |
     3
 4 | (system user)                               |          0 | assign |
     4
 5 | (super user) and (query group: superuser)   |          0 | assign |
     5
 6 | (query group: querygroup1)                  |          0 | assign |
     6
 7 | (user group: usergroup1)                    |          0 | assign |
     6
 8 | (user group: usergroup2)                    |          0 | assign |
     7
 9 | (query group: querygroup3)                  |          0 | assign |
     8
10 | (query group: querygroup4)                  |          0 | assign |
     9
11 | (user group: usergroup4)                    |          0 | assign |
     9
12 | (query group: querygroup*)                  |          0 | assign |
    10
13 | (user group: usergroup*)                    |          0 | assign |
    10
14 | (querytype: any)                            |          0 | assign |
    11
(4 rows)
```

STV_WLM_QMR_CONFIG

Enregistre la configuration des règles de surveillance de requête (QRM) WLM. Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.

STV_WLM_QMR_CONFIG n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
service_class	entier	ID de la file d'attente de requêtes WLM (classe de service). Les files d'attente de requêtes sont définies dans la configuration WLM. Des règles peuvent être définies uniquement pour les files d'attente définies par l'utilisateur. Pour obtenir la liste des classes de service IDs, consultez Classe de service WLM IDs.
rule_name	character(256)	Nom de la règle de surveillance de requête.
action	character(256)	Action de la règle. Les valeurs possibles sont `log`, `hop`, `abort` et `change_query_priority` .
metric_name	character(256)	Nom de la métrique.
metric_operator	character(256)	Opérateur de la métrique. Les valeurs possibles sont >, =, <.
metric_value	double	La valeur de seuil pour la métrique spécifiée déclenchant une action.
action_value	character(256)	Si `action` est `change_query_priority` , les valeurs possibles sont `highest`, `high`, `normal`, `low` et `lowest`. Si `action` est `log`, `hop` ou `abort`, la valeur est vide.

Exemple de requête

Pour afficher les définitions des règles de surveillance de requête (QRM) pour toutes les classes de service supérieures à 5 (ce qui inclut les files d'attente définies par l'utilisateur), exécutez la requête ci-après. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.

```
Select *
from stv_wlm_qmr_config
where service_class > 5
```

```
order by service_class;
```

STV_WLM_QUERY_QUEUE_STATE

Enregistre l'état actuel des files d'attente de requête pour les classes de service.

STV_WLM_QUERY_QUEUE_STATE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
service_class	entier	ID de la classe de service. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.
position	entier	Position de la requête dans la file d'attente. La requête avec la valeur de **position** la plus petite s'exécute ensuite.
tâche	entier	ID utilisé pour suivre une requête via le gestionnaire de la charge de travail. Peut être associé à plusieurs requêtes IDs. Si une requête est redémarrée, la requête se voit attribuer un nouvel ID de requête, mais pas un nouvel ID de tâche.
query	entier	ID de requête. Si une requête est redémarrée, la requête se voit attribuer un nouvel ID de requête, mais pas un nouvel ID de tâche.
slot_count	entier	Nombre d'emplacements de requête WLM.

Nom de la colonne	Type de données	Description
start_time	timestamp	Heure à laquelle la requête est entrée dans la file d'attente.
queue_time	bigint	Nombre de microsecondes pendant lesquelles la requête s'est trouvée dans la file d'attente.

Exemple de requête

La requête suivante affiche les requêtes se trouvant dans la file d'attente pour des classes de service supérieures à 4.

```
select * from stv_wlm_query_queue_state
where service_class > 4
order by service_class;
```

Cette requête renvoie l'exemple de sortie suivant.

```
 service_class | position | task | query | slot_count |        start_time          |
 queue_time
---------------+----------+------+-------+------------+----------------------------
+------------
             5 |        0 |  455 |   476 |          5 | 2010-10-06 13:18:24.065838 |
 20937257
             6 |        1 |  456 |   478 |          5 | 2010-10-06 13:18:26.652906 |
 18350191
(2 rows)
```

STV_WLM_QUERY_STATE

Enregistre l'état actuel des requêtes en cours suivies par WLM.

STV_WLM_QUERY_STATE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
xid	entier	ID de transaction de la requête ou sous-requête.
tâche	entier	ID utilisé pour suivre une requête via le gestionnaire de la charge de travail. Peut être associé à plusieurs requêtes IDs. Si une requête est redémarrée, la requête se voit attribuer un nouvel ID de requête, mais pas un nouvel ID de tâche.
query	entier	ID de requête. Si une requête est redémarrée, la requête se voit attribuer un nouvel ID de requête, mais pas un nouvel ID de tâche.
service_class	entier	ID de la classe de service. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.
slot_count	entier	Nombre d'emplacements de requête WLM.
wlm_start_time	timestamp	Heure à laquelle la requête est entrée dans la file d'attente de la table système ou de la courte file d'attente de la requête.
state	character(16)	État actuel de la requête ou sous-requête. Les valeurs possibles sont les suivantes : • `Classified` – La requête a été affectée à une classe de service. • `Completed` – L'exécution de la requête est terminée. La requête s'est exécutée correctement ou a été

Nom de la colonne	Type de données	Description
		annulée. Pour connaître l'état final, vérifiez les résultats de STL_QUERY. • `Dequeued` – Utilisation interne uniquement. • `Evicted` – La requête a été expulsée de la classe de service pour le redémarrage. • `Evicting` – La requête est expulsée de la classe de service pour le redémarrage. • `Initialized` – Utilisation interne uniquement. • `Invalid` – Utilisation interne uniquement. • `Queued` – La requête a été envoyée à la file d'attente de requêtes, car aucun emplacement n'était disponible pour l'exécuter. • `QueuedWaiting` – La requête est en attente dans la file d'attente de requêtes. • `Rejected` – Utilisation interne uniquement. • `Returning` – La requête renvoie des résultats au client. • `Running` – La requête est en cours d'exécution. • `TaskAssigned` – Utilisation interne uniquement.
queue_time	bigint	Nombre de microsecondes que la requête a passé dans la file d'attente.
exec_time	bigint	Nombre de microsecondes pendant lesquelles la requête était en cours d'exécution.
query_priority	char(20)	Priorité de la requête. Les valeurs possibles sont `n/a`, `lowest`, `low`, `normal`, `high` et `highest`, où `n/a` signifie que cette priorité de requête n'est pas prise en charge.

Exemple de requête

La requête suivante affiche toutes les requêtes en cours d'exécution dans les classes de service supérieures à 4. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.

```
select xid, query, trim(state) as state, queue_time, exec_time
from stv_wlm_query_state
where service_class > 4;
```

Cette requête renvoie l'exemple de sortie suivant :

```
xid    | query | state   | queue_time | exec_time
-------+-------+---------+------------+-----------
100813 | 25942 | Running |          0 |    1369029
100074 | 25775 | Running |          0 | 2221589242
```

STV_WLM_QUERY_TASK_STATE

Contient l'état actuel des tâches de requête de classe de service.

STV_WLM_QUERY_TASK_STATE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
service_class	entier	ID de la classe de service. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.
tâche	entier	ID utilisé pour suivre une requête via le gestionnaire de la charge de travail. Peut être associé à plusieurs requêtes IDs. Si une requête est redémarrée, la requête se voit attribuer un nouvel ID de requête, mais pas un nouvel ID de tâche.

Nom de la colonne	Type de données	Description
query	entier	ID de requête. Si une requête est redémarrée, la requête se voit attribuer un nouvel ID de requête, mais pas un nouvel ID de tâche.
slot_count	entier	Nombre d'emplacements de requête WLM.
start_time	timestamp	Heure à laquelle la requête a commencé à s'exécuter.
exec_time	bigint	Nombre de microsecondes pendant lesquelles la requête s'est exécutée.

Exemple de requête

La requête suivante affiche l'état actuel des requêtes dans les classes de service supérieures à 4. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.

```
select * from stv_wlm_query_task_state
where service_class > 4;
```

Cette requête renvoie l'exemple de sortie suivant :

```
service_class | task | query |         start_time         | exec_time
--------------+------+-------+----------------------------+-----------
    5         |  466 |   491 | 2010-10-06 13:29:23.063787 | 357618748
(1 row)
```

STV_WLM_SERVICE_CLASS_CONFIG

Enregistre les configurations de classe de service pour WLM.

STV_WLM_SERVICE_CLASS_CONFIG n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
service_class	entier	ID de la classe de service. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.
queueing_strategy	character(32)	Réservé au système.
num_query_tasks	entier	Niveau de simultanéité réel en vigueur de la classe de service. Si les colonnes `num_query_tasks` et `target_num_query_tasks` sont différentes, une transition WLM dynamique est en cours. La valeur `-1` indique que l'option Auto WLM (WLM auto) est configurée.
target_num_query_tasks	entier	Niveau de simultanéité défini par la modification de configuration WLM la plus récente.
evictable	character(8)	Réservé au système.
eviction_threshold	bigint	Réservé au système.
query_working_mem	entier	Quantité de mémoire de travail réelle actuelle, en Mo par emplacement, par nœud, affectée à la classe de service. Si les colonnes `query_working_mem` et `target_query_working_mem` sont différentes, une transition WLM dynamique est en cours. La valeur `-1` indique que l'option Auto WLM (WLM auto) est configurée.
target_query_working_mem	entier	Quantité de mémoire de travail, en Mo par emplacement, par nœud, définie par la modification de configuration WLM la plus récente.
min_step_mem	entier	Réservé au système.
name	character(64)	Nom de la classe de service.

Nom de la colonne	Type de données	Description
max_execution_time	bigint	Nombre de millisecondes pendant lesquelles la requête peut s'exécuter avant d'être résiliée.
user_group_wild_card	Booléen	Si TRUE, la file d'attente WLM traite un astérisque (*) en tant que caractère générique dans les chaînes de groupe d'utilisateurs dans la configuration WLM.
query_group_wild_card	Booléen	Si TRUE, la file d'attente WLM traite un astérisque (*) en tant que caractère générique dans les chaînes de groupe de requêtes dans la configuration WLM.
concurrency_scaling	character(20)	Décrit si la mise à l'échelle de la concurrence est on ou `off`.
query_priority	character(20)	La valeur de la priorité de la requête.
user_role_wild_card	Booléen	Si TRUE, la file d'attente WLM traite un astérisque (*) en tant que caractère générique dans les chaînes d'utilisateurs dans la configuration WLM.

Exemple de requête

La première classe de service définie par l'utilisateur est la classe de service 6, qui est appelée la classe de service n° 1. La requête suivante affiche la configuration actuelle des classes de service supérieures à 4. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.

```
select rtrim(name) as name,
num_query_tasks as slots,
query_working_mem as mem,
max_execution_time as max_time,
user_group_wild_card as user_wildcard,
query_group_wild_card as query_wildcard
from stv_wlm_service_class_config
where service_class > 4;
```

```
name                          | slots | mem | max_time | user_wildcard | query_wildcard
------------------------------+-------+-----+----------+---------------+---------------
Service class for super user |     1 | 535 |        0 | false         | false
Queue 1                       |     5 | 125 |        0 | false         | false
Queue 2                       |     5 | 125 |        0 | false         | false
Queue 3                       |     5 | 125 |        0 | false         | false
Queue 4                       |     5 | 627 |        0 | false         | false
Queue 5                       |     5 | 125 |        0 | true          | true
Default queue                 |     5 | 125 |        0 | false         | false
```

La requête suivante affiche l'état d'une transition WLM dynamique. Pendant que la transition est en cours, `num_query_tasks` et `target_query_working_mem` sont mises à jour jusqu'à ce qu'elles soient égales aux valeurs cibles. Pour de plus amples informations, veuillez consulter Propriétés de configuration dynamiques et statiques WLM.

```
select rtrim(name) as name,
num_query_tasks as slots,
target_num_query_tasks as target_slots,
query_working_mem as memory,
target_query_working_mem as target_memory
from stv_wlm_service_class_config
where num_query_tasks > target_num_query_tasks
or query_working_mem > target_query_working_mem
and service_class > 5;

 name             | slots | target_slots | memory | target_mem
------------------+-------+--------------+--------+------------
 Queue 3          |     5 |           15 |    125 |       375
 Queue 5          |    10 |            5 |    250 |       125
 (2 rows)
```

STV_WLM_SERVICE_CLASS_STATE

Contient l'état actuel des classes de service.

STV_WLM_SERVICE_CLASS_STATE n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
service_class	entier	ID de la classe de service. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.
num_queued_queries	entier	Nombre de requêtes actuellement dans la file d'attente.
num_executing_queries	entier	Nombre de requêtes en cours d'exécution.
num_serviced_queries	entier	Nombre de requêtes s'étant déjà trouvées dans la classe de service.
num_executed_queries	entier	Nombre de requêtes qui ont été exécutées depuis le redémarrage d'Amazon Redshift.
num_evicted_queries	entier	Nombre de requêtes qui ont été évincées depuis le redémarrage d'Amazon Redshift. Une requête peut être expulsée en raison de l'expiration du délai WLM, d'une action de saut de règle de surveillance de requête (QMR) ou d'un échec de la requête sur un cluster de mise à l'échelle de la simultanéité.
num_concurrency_scaling_queries	entier	Nombre de requêtes exécutées sur un cluster de mise à l'échelle de la concurrence depuis le redémarrage d'Amazon Redshift.

Exemple de requête

La requête suivante affiche l'état des classes de service supérieures à 5. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.

```
select service_class, num_executing_queries,
num_executed_queries
from stv_wlm_service_class_state
where service_class > 5
```

```
order by service_class;
```

```
 service_class | num_executing_queries | num_executed_queries
---------------+-----------------------+----------------------
             6 |                     1 |                  222
             7 |                     0 |                  135
             8 |                     1 |                   39
(3 rows)
```

STV_XRESTORE_ALTER_QUEUE_STATE

Utilisez STV_XRESTORE_ALTER_QUEUE_STATE pour suivre la progression de la migration de chaque table lors d'un redimensionnement classique. Cela s'applique spécifiquement lorsque le type de nœud cible est RA3. Pour plus d'informations sur le redimensionnement classique en RA3 nœuds, consultez la section Redimensionnement classique.

STV_XRESTORE_ALTER_QUEUE_STATE n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_RESTORE_STATE. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a initié le redimensionnement.
db_id	entier	ID de la base de données.
schema	char(128)	Nom du schéma.
table_name	char(128)	Nom de la table.
tbl	entier	ID de la table.
status	char(64)	État de la progression de la migration de la table. Les valeurs possibles sont les suivantes.

Nom de la colonne	Type de données	Description
		• `Waiting`: En attente du début de la redistribution • `Applying`: En cours de redistribution • `Finished`: Redistribution terminée
type_tâche	entier	Type de redistribution de la table. Les valeurs possibles sont les suivantes. • 1: CLÉ • 2: MÊME Pour plus d'informations sur les styles de distribution, consultez Styles de distribution.

Exemple de requête

La requête suivante indique le nombre de tables d'une base de données qui attendent d'être redimensionnées, qui sont en cours de redimensionnement et dont le redimensionnement est terminé.

```
select db_id, status, count(*)
from stv_xrestore_alter_queue_state
group by 1,2 order by 3 desc

db_id  |   status   | count
-------+------------+------
694325 | Waiting    |   323
694325 | Finished   |    60
694325 | Applying   |     1
```

Vues SVCS pour le cluster principal et les clusters de mise à l'échelle de la simultanéité

Les vues système SVCS dotées du préfixe SVCS fournissent des détails sur les requêtes exécutées sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires aux

tables dotées du préfixe STL, si ce n'est que les tables STL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

Rubriques

- SVCS_ALERT_EVENT_LOG
- SVCS_COMPILE
- SVCS_CONCURRENCY_SCALING_USAGE
- SVCS_EXPLAIN
- SVCS_PLAN_INFO
- SVCS_QUERY_SUMMARY
- SVCS_S3LIST
- SVCS_S3LOG
- SVCS_S3PARTITION_SUMMARY
- SVCS_S3QUERY_SUMMARY
- SVCS_STREAM_SEGS
- SVCS_UNLOAD_LOG

SVCS_ALERT_EVENT_LOG

Enregistre une alerte quand l'optimiseur de requête identifie les conditions qui peuvent indiquer des problèmes de performances. Cette vue est dérivée de la table système STL_ALERT_EVENT_LOG, mais n'affiche pas le niveau tranche pour les requêtes exécutées sur un cluster de mise à l'échelle de la simultanéité. Utilisez la table SVCS_ALERT_EVENT_LOG pour identifier les opportunités d'améliorer les performances des requêtes.

Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Pour de plus amples informations, veuillez consulter Traitement des requêtes.

> **Note**
>
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires aux tables dotées du préfixe STL, si ce n'est que les tables STL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_ALERT_EVENT_LOG est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Étape de la requête exécutée.
pid	entier	ID de processus associé à l'instruction et à la tranche. La même requête peut en avoir plusieurs PIDs si elle s'exécute sur plusieurs tranches.
xid	bigint	ID de transaction associé à l'instruction.
event	character (1024)	Description de l'événement d'alerte.
solution	character (1024)	Solution recommandée.
event_time	timestamp	Heure au format UTC du début de la requête. Le temps total inclut la mise en file d'attente et l'exécution avec 6 chiffres de précision pour les fractions de secondes. olpPar exemple : **`2009-06-12 11:29:19.131358`** .

Notes d'utilisation

Vous pouvez utiliser SVCS_ALERT_EVENT_LOG pour identifier les problèmes potentiels de votre requête, puis suivre les pratiques décrites dans Réglage des performances des

requêtes afin d'optimiser la conception de votre base de données et de réécrire vos requêtes. SVCS_ALERT_EVENT_LOG enregistre les alertes suivantes :

- Statistiques manquantes

  Il manque des statistiques. Exécutez ANALYZE après les chargements de données ou les mises à jour importantes, et utilisez STATUPDATE avec les opérations COPY. Pour plus d'informations, consultez Bonnes pratiques Amazon Redshift pour la conception de requêtes.

- Boucle imbriquée

  Une boucle imbriquée est généralement un produit cartésien. Évaluez votre requête afin de vous assurer que toutes les tables participantes sont unies efficacement.

- Filtre très sélectif

  Le rapport entre le nombre de lignes retournées et le nombre de lignes analysées est inférieur à 0,05. La valeur des lignes analysées est `rows_pre_user_filter` et celle des lignes retournées la valeur des lignes dans la table système STL_SCAN. Indique que la requête analyse un nombre inhabituellement grand de lignes pour déterminer le jeu de résultats. La raison peut en être des clés de tri manquantes ou incorrectes. Pour plus d'informations, consultez Clés de tri.

- Lignes fantôme excessives

  Une analyse a ignoré un nombre relativement grand de lignes marquées comme supprimées, mais pas vidées, ou de lignes qui ont été insérées, mais pas validées. Pour plus d'informations, consultez Exécution de l'opération VACUUM sur les tables.

- Grande Distribution

  Plus de 1 000 000 de lignes ont été redistribuées pour une jointure par hachage ou une agrégation. Pour plus d'informations, consultez Distribution des données pour l'optimisation des requêtes.

- Large diffusion

  Plus de 1 000 000 de lignes ont été diffusées pour une jointure par hachage. Pour plus d'informations, consultez Distribution des données pour l'optimisation des requêtes.

- Exécution en série

  Un style de redistribution DS_DIST_ALL_INNER a été indiqué dans le plan de requête, ce qui force une exécution en série, car la totalité de la table interne a été redistribuée sur un seul nœud. Pour plus d'informations, consultez Distribution des données pour l'optimisation des requêtes.

Exemples de requêtes

La requête suivante affiche les événements d'alerte pour quatre requêtes.

```
SELECT query, substring(event,0,25) as event,
substring(solution,0,25) as solution,
trim(event_time) as event_time from svcs_alert_event_log order by query;

 query |             event             |          solution            |     event_time

-------+-------------------------------+------------------------------
+---------------------
  6567 | Missing query planner statist | Run the ANALYZE command      | 2014-01-03
 18:20:58
  7450 | Scanned a large number of del | Run the VACUUM command to rec| 2014-01-03
 21:19:31
  8406 | Nested Loop Join in the query | Review the join predicates to| 2014-01-04
 00:34:22
 29512 | Very selective query filter:r | Review the choice of sort key| 2014-01-06
 22:00:00

(4 rows)
```

SVCS_COMPILE

Les enregistrements compilent le temps et l'emplacement pour chaque segment de requête, y compris les requêtes exécutées sur un cluster de mise à l'échelle et les requêtes exécutées sur le cluster principal.

> **Note**
>
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires aux vues dotées du préfixe SVL, si ce n'est que les vues SVL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_COMPILE est visible de tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Pour plus d'informations sur SCL_COMPILE, consultez SVL_COMPILE.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
xid	bigint	ID de transaction associé à l'instruction.
pid	entier	ID de processus associé à l'instruction.
query	entier	ID de requête. Vous pouvez utiliser cet ID pour joindre d'autres vues et tables système.
segment	entier	Segment de la requête à compiler.
locus	entier	Emplacement où le segment s'exécute. **1** s'il s'exécute sur un nœud de calcul et **2** s'il s'exécute sur le nœud principal.
starttime	timestamp	Heure UTC (Universal Coordinated Time) à laquelle la compilation a commencé.
endtime	timestamp	Heure UTC (Universal Coordinated Time) à laquelle la compilation a pris fin.
compile	entier	Valeur **0** si la compilation a été réutilisée, ou **1** si le segment a été compilé.

Exemples de requêtes

Dans cet exemple, les requêtes 35878 et 35879 ont exécuté la même instruction SQL. La colonne de compilation de la requête 35878 affiche 1 pour quatre segments de requête, ce qui indique que les segments ont été compilés. La requête 35879 affiche `0` dans la colonne de compilation de chaque segment, ce qui indique que les segments n'avaient pas besoin d'être compilés à nouveau.

```
select userid, xid,  pid, query, segment, locus,
datediff(ms, starttime, endtime) as duration, compile
from svcs_compile
```

```
where query = 35878 or query = 35879
order by query, segment;

 userid |  xid   |  pid  | query | segment | locus | duration | compile
--------+--------+-------+-------+---------+-------+----------+---------
    100 | 112780 | 23028 | 35878 |       0 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       1 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       2 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       3 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       4 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       5 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       6 |     1 |     1380 |       1
    100 | 112780 | 23028 | 35878 |       7 |     1 |     1085 |       1
    100 | 112780 | 23028 | 35878 |       8 |     1 |     1197 |       1
    100 | 112780 | 23028 | 35878 |       9 |     2 |      905 |       1
    100 | 112782 | 23028 | 35879 |       0 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       1 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       2 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       3 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       4 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       5 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       6 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       7 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       8 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       9 |     2 |        0 |       0
(20 rows)
```

SVCS_CONCURRENCY_SCALING_USAGE

Enregistre les périodes d'utilisation de la mise à l'échelle de la simultanéité. Chaque période d'utilisation est une durée ininterrompue au cours de laquelle un cluster de mise à l'échelle de la simultanéité traite activement des requêtes.

SVCS_CONCURRENCY_SCALING_USAGE Cette table est visible par les super-utilisateurs. Le super-utilisateur de la base de données peut choisir de l'ouvrir à tous les utilisateurs.

Colonnes de la table

Nom de la colonne	Type de données	Description
start_time	horodatage sans fuseau horaire	Au démarrage de la période d'utilisation.
end_time	horodatage sans fuseau horaire	À la fin de la période d'utilisation.
queries	bigint	Nombre de requêtes exécutées pendant cette période d'utilisation.
usage_in_seconds	numeric(27,0)	Nombre total de secondes de cette période d'utilisation.

Exemples de requêtes

Pour afficher la durée d'utilisation en secondes pour une période spécifique, entrez la requête suivante :

```
select * from svcs_concurrency_scaling_usage order by start_time;

start_time | end_time | queries | usage_in_seconds
---------------------------+---------------------------+---------+-----------------
2019-02-14 18:43:53.01063 | 2019-02-14 19:16:49.781649 | 48 | 1977
```

SVCS_EXPLAIN

Affiche le plan EXPLAIN pour une requête qui a été soumise à exécution.

> **Note**
>
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires

aux tables dotées du préfixe STL, si ce n'est que les tables STL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_EXPLAIN est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
nodeid	entier	Identificateur de nœud de plan, où un nœud correspond à une ou plusieurs étapes de l'exécution de la requête.
parentid	entier	Identificateur de nœud de plan d'un nœud parent. Un nœud parent a un certain nombre de nœuds enfants. Par exemple, une jointure de fusion est le parent des analyses des tables jointes.
plannode	character(400)	Texte du nœud de la sortie EXPLAIN. Les nœuds de plan qui font référence à l'exécution sur les nœuds de calcul sont préfixés par **XN** dans la sortie EXPLAIN.
info	character(400)	Informations de qualificateur et de filtre pour le nœud de plan. Par exemple, les conditions de jointure et les restrictions clause WHERE sont incluses dans cette colonne.

Exemples de requêtes

Considérons la sortie EXPLAIN suivante d'une requête de jointure d'agrégation :

```
explain select avg(datediff(day, listtime, saletime)) as avgwait
```

```
from sales, listing where sales.listid = listing.listid;
                                QUERY PLAN

------------------------------------------------------------------------------
 XN Aggregate  (cost=6350.30..6350.31 rows=1 width=16)
  ->  XN Hash Join DS_DIST_NONE  (cost=47.08..6340.89 rows=3766 width=16)
        Hash Cond: ("outer".listid = "inner".listid)
        -> XN Seq Scan on listing  (cost=0.00..1924.97 rows=192497 width=12)
        -> XN Hash  (cost=37.66..37.66 rows=3766 width=12)
              -> XN Seq Scan on sales  (cost=0.00..37.66 rows=3766 width=12)
(6 rows)
```

Si vous exécutez cette requête et que son ID de requête est 10, vous pouvez utiliser la table SVCS_EXPLAIN pour afficher le même type d'informations que renvoie la commande EXPLAIN :

```
select query,nodeid,parentid,substring(plannode from 1 for 30),
substring(info from 1 for 20) from svcs_explain
where query=10 order by 1,2;

query| nodeid |parentid|           substring            |    substring
-----+--------+--------+--------------------------------+-------------------
10   |      1 |      0 |XN Aggregate  (cost=6717.61..6  |
10   |      2 |      1 |  -> XN Merge Join DS_DIST_NO| Merge Cond:("outer"
10   |      3 |      2 |       -> XN Seq Scan on lis |
10   |      4 |      2 |       -> XN Seq Scan on sal |
(4 rows)
```

Considérons la requête suivante :

```
select event.eventid, sum(pricepaid)
from event, sales
where event.eventid=sales.eventid
group by event.eventid order by 2 desc;

eventid |   sum
--------+----------
    289 | 51846.00
   7895 | 51049.00
   1602 | 50301.00
    851 | 49956.00
   7315 | 49823.00
...
```

Si l'ID de requête est le 15, la requête de la table système suivante renvoie les nœuds de plan qui ont été exécutés. Dans ce cas, l'ordre des nœuds est inversé pour afficher l'ordre réel d'exécution :

```
select query,nodeid,parentid,substring(plannode from 1 for 56)
from svcs_explain where query=15 order by 1, 2 desc;

query|nodeid|parentid|                          substring
-----+------+--------+-----------------------------------------------------
15   |    8 |      7 |                               -> XN Seq Scan on eve
15   |    7 |      5 |                          -> XN Hash(cost=87.98..87.9
15   |    6 |      5 |                          -> XN Seq Scan on sales(cos
15   |    5 |      4 |                    -> XN Hash Join DS_DIST_OUTER(cos
15   |    4 |      3 |              -> XN HashAggregate(cost=862286577.07..
15   |    3 |      2 |        -> XN Sort(cost=1000862287175.47..10008622871
15   |    2 |      1 |  -> XN Network(cost=1000862287175.47..1000862287197.
15   |    1 |      0 |XN Merge(cost=1000862287175.47..1000862287197.46 rows=87
(8 rows)
```

La requête suivante extrait la requête IDs pour tous les plans de requête contenant une fonction de fenêtre :

```
select query, trim(plannode) from svcs_explain
where plannode like '%Window%';

query|                                  btrim
-----+-------------------------------------------------------------------------
26   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
27   | -> XN Window(cost=1000985348268.57..1000985351256.98 rows=170 width=33)
(2 rows)
```

SVCS_PLAN_INFO

Utilisez la table SVCS_PLAN_INFO pour examiner la sortie EXPLAIN d'une requête en termes d'ensemble de lignes. Il s'agit d'un autre moyen de regarder les plans de requête.

> **Note**
>
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires

aux tables dotées du préfixe STL, si ce n'est que les tables STL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_PLAN_INFO est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
nodeid	entier	Identificateur de nœud de plan, où un nœud correspond à une ou plusieurs étapes de l'exécution de la requête.
segment	entier	Numéro qui identifie le segment de requête.
étape	entier	Numéro qui identifie l'étape de la requête.
locus	entier	Emplacement où l'étape s'exécute. 0 si sur un nœud de calcul et 1 si sur le nœud principal.
plannode	entier	Valeur énumérée du nœud de plan. Consultez le tableau suivant pour obtenir les enums pour plannode. (La colonne PLANNODE de SVCS_EXPLAIN contient le texte du nœud de plan.)
startupcost	double precision	Coût relatif estimé du retour de la première ligne pour cette étape.
totalcost	double precision	Coût estimé relatif de l'exécution de l'étape.
rows	bigint	Nombre estimé de lignes qui seront produites par l'étape.

Nom de la colonne	Type de données	Description
octets	bigint	Nombre estimé d'octets qui seront produits par l'étape.

Exemples de requêtes

Les exemples suivants comparent les plans de requête d'une simple requête SELECT retournée à l'aide de la commande EXPLAIN et de l'interrogation de la table SVCS_PLAN_INFO.

```
explain select * from category;
QUERY PLAN
--------------------------------------------------------------
XN Seq Scan on category (cost=0.00..0.11 rows=11 width=49)
(1 row)

select * from category;
catid | catgroup | catname | catdesc
-------+----------+-----------+--------------------------------------------
1 | Sports | MLB | Major League Baseball
3 | Sports | NFL | National Football League
5 | Sports | MLS | Major League Soccer
...

select * from svcs_plan_info where query=256;

query | nodeid | segment | step | locus | plannode | startupcost | totalcost
| rows | bytes
-------+--------+---------+------+-------+----------+-------------+-----------+------
+-------
256 | 1 | 0 | 1 | 0 | 104 | 0 | 0.11 | 11 | 539
256 | 1 | 0 | 0 | 0 | 104 | 0 | 0.11 | 11 | 539
(2 rows)
```

Dans cet exemple, PLANNODE 104 fait référence à l'analyse séquentielle de la table CATEGORY.

```
select distinct eventname from event order by 1;

eventname
----------------------------------------------------------------------
.38 Special
```

```
3 Doors Down
70s Soul Jam
A Bronx Tale
...

explain select distinct eventname from event order by 1;

QUERY PLAN
------------------------------------------------------------------------------
XN Merge (cost=1000000000136.38..1000000000137.82 rows=576 width=17)
Merge Key: eventname
-> XN Network (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Send to leader
-> XN Sort (cost=1000000000136.38..1000000000137.82 rows=576
width=17)
Sort Key: eventname
-> XN Unique (cost=0.00..109.98 rows=576 width=17)
-> XN Seq Scan on event (cost=0.00..87.98 rows=8798
width=17)
(8 rows)

select * from svcs_plan_info where query=240 order by nodeid desc;

query | nodeid | segment | step | locus | plannode | startupcost |
totalcost | rows | bytes
-------+--------+---------+------+-------+----------+-----------------
+------------------+------+--------
240 | 5 | 0 | 0 | 0 | 104 | 0                | 87.98   | 8798 | 149566
240 | 5 | 0 | 1 | 0 | 104 | 0                | 87.98   | 8798 | 149566
240 | 4 | 0 | 2 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 0 | 3 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 1 | 0 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 4 | 1 | 1 | 0 | 117 | 0                | 109.975 | 576  | 9792
240 | 3 | 1 | 2 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 3 | 2 | 0 | 0 | 114 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 2 | 2 | 1 | 0 | 123 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
240 | 1 | 3 | 0 | 0 | 122 | 1000000000136.38 | 1000000000137.82 | 576 | 9792
(10 rows)
```

SVCS_QUERY_SUMMARY

Utilisez la vue SVCS_QUERY_SUMMARY pour rechercher des informations générales sur l'exécution d'une requête.

Notez que les informations de SVCS_QUERY_SUMMARY sont regroupées à partir de tous les nœuds.

> **Note**
>
> La vue SVCS_QUERY_SUMMARY contient uniquement des informations sur les requêtes exécutées par Amazon Redshift, et non sur d'autres commandes DDL ou d'utilitaire. Pour obtenir une liste complète et des informations sur toutes les instructions exécutées par Amazon Redshift, y compris les commandes DDL et d'utilitaire, vous pouvez interroger la vue SVL_STATEMENTTEXT.
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires aux vues dotées du préfixe SVL, si ce n'est que les vues SVL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_QUERY_SUMMARY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Pour plus d'informations sur SVL_QUERY_SOMMY, consultez SVL_QUERY_SUMMARY.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.

Nom de la colonne	Type de données	Description
query	entier	ID de requête. Permet de joindre d'autres tables système et vues.
stm	entier	Flux : ensemble de segments simultanés d'une requête. Une requête possède un ou plusieurs flux.
seg	entier	Numéro de segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Les segments de requête peuvent s'exécuter en parallèle. Chaque segment s'exécute dans un processus unique.
étape	entier	Étape de la requête exécutée.
maxtime	bigint	Durée maximale (en microsecondes) d'exécution de l'étape.
avgtime	bigint	Durée moyenne (en microsecondes) d'exécution de l'étape.
rows	bigint	Nombre de lignes de données impliquées dans l'étape de requête.
bytes	bigint	Nombre d'octets de données impliqués dans l'étape de requête.
rate_row	double precision	Taux d'exécution de la requête par ligne.
rate_byte	double precision	Taux d'exécution de la requête par octet.
étiquette	text	Étiquette de l'étape, qui se compose d'un nom d'étape de requête et, le cas échéant, d'un ID de table et d'un nom de table (par exemple, scan tbl=100448 name =user). Les tables à trois chiffres font IDs généralement référence à des scans de tables transitoires. Lorsque `tbl=0` s'affiche, cela fait généralement référence à une analyse d'une valeur constante.

Nom de la colonne	Type de données	Description
is_diskbased	character(1)	Indique si cette étape de la requête a été exécutée comme opération sur disque sur un nœud du cluster : true (**t**) ou false (**f**). Seules certaines étapes, telles que le hachage, le tri et l'agrégation, peuvent accéder au disque. La plupart des types d'étapes sont toujours exécutés en mémoire.
workmem	bigint	Quantité de mémoire de travail (en octets) attribuée à l'étape de requête.
is_rrscan	character(1)	Si la valeur est définie sur true (**t**), indique qu'une analyse à plage restreinte a été utilisée sur l'étape. La valeur par défaut est false (**f**).
is_delayed_scan	character(1)	Si la valeur est définie sur true (**t**), indique qu'une analyse retardée a été utilisée sur l'étape. La valeur par défaut est false (**f**).
rows_pre_filter	bigint	Pour les analyses des tables permanentes, le nombre total de lignes émises avant le filtrage des lignes marquées comme devant être supprimées (lignes fantôme).

Exemples de requêtes

Affichage des informations de traitement d'une étape de requête

La requête suivante affiche les informations de traitement de base de chaque étape de la requête 87 :

```
select query, stm, seg, step, rows, bytes
from svcs_query_summary
where query = 87
order by query, seg, step;
```

Cette requête extrait les informations de traitement de la requête 87, comme illustré dans l'exemple de sortie suivant :

```
 query | stm | seg | step |  rows  |  bytes
-------+-----+-----+------+--------+---------
```

```
87     |   0 |   0 |    0 |     90 |    1890
87     |   0 |   0 |    2 |     90 |     360
87     |   0 |   1 |    0 |     90 |     360
87     |   0 |   1 |    2 |     90 |    1440
87     |   1 |   2 |    0 | 210494 | 4209880
87     |   1 |   2 |    3 |  89500 |       0
87     |   1 |   2 |    6 |      4 |      96
87     |   2 |   3 |    0 |      4 |      96
87     |   2 |   3 |    1 |      4 |      96
87     |   2 |   4 |    0 |      4 |      96
87     |   2 |   4 |    1 |      1 |      24
87     |   3 |   5 |    0 |      1 |      24
87     |   3 |   5 |    4 |      0 |       0
(13 rows)
```

Détermination si les étapes de requête ont été répandues sur le disque

La requête suivante indique si l'une des étapes de la requête avec l'ID de requête 1025 (voir la vue SVL_QLOG pour apprendre à obtenir l'ID d'une requête) a été répandue sur le disque ou si la requête a été entièrement exécutée en mémoire :

```
select query, step, rows, workmem, label, is_diskbased
from svcs_query_summary
where query = 1025
order by workmem desc;
```

Cette requête renvoie l'exemple de sortie suivant :

```
query| step|  rows  |  workmem   |  label        | is_diskbased
-----+-----+--------+------------+---------------+--------------
1025 |  0  |16000000|  141557760 |scan tbl=9     | f
1025 |  2  |16000000|  135266304 |hash tbl=142   | t
1025 |  0  |16000000|  128974848 |scan tbl=116536| f
1025 |  2  |16000000|  122683392 |dist           | f
(4 rows)
```

En analysant les valeurs d'IS_DISKBASED, vous pouvez voir quelles étapes de requête sont allées sur le disque. Pour la requête 1025, l'étape de hachage a été exécutée sur le disque. Les étapes susceptibles de s'exécuter sur disque incluent les étapes de hachage, d'agrégation et de tri. Pour afficher uniquement les étapes de requête sur disque, ajoutez la clause **`and is_diskbased = 't'`** à l'instruction SQL de l'exemple ci-dessus.

SVCS_S3LIST

Utilisez la vue SVCS_S3LIST afin d'obtenir des détails sur les requêtes Amazon Redshift Spectrum au niveau du segment. Un segment peut exécuter une analyse de table externe. Cette vue est dérivée de la vue système SVL_S3LIST, mais n'affiche pas le niveau tranche pour les requêtes exécutées sur un cluster de mise à l'échelle de la simultanéité.

> **Note**
>
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires aux vues dotées du préfixe SVL, si ce n'est que les vues SVL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_S3LIST est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Pour plus d'informations sur SVL_S3LIST, consultez SVL_S3LIST.

Colonnes de la table

Nom de la colonne	Type de données	Description
query	entier	ID de requête.
segment	entier	Numéro du segment. Requête composée de plusieurs segments.
node	entier	Numéro du nœud.
eventtime	timestamp	L'heure UTC à laquelle l'événement est enregistré.
bucket	char(256)	Nom du compartiment Amazon S3.
prefix	char(256)	Préfixe de l'emplacement du compartiment Amazon S3.
recursive	char(1)	Si l'analyse des sous-dossiers est récursive.

Nom de la colonne	Type de données	Description
retrieved_files	entier	Nombre de fichiers répertoriés.
max_file_size	bigint	Taille de fichier maximum parmi les fichiers répertoriés.
avg_file_size	double precision	Taille de fichier moyenne parmi les fichiers répertoriés.
generated_splits	entier	Nombre de fichiers séparés.
avg_split_length	double precision	Longueur moyenne des fichiers séparés en octets.
duration	bigint	Durée de l'établissement de la liste de fichiers en microsecondes.

Exemple de requête

L'exemple suivant interroge SVCS_S3LIST par rapport à la dernière requête exécutée.

```
select *
from svcs_s3list
where query = pg_last_query_id()
order by query,segment;
```

SVCS_S3LOG

Utilisez la vue SVCS_S3LOG afin d'obtenir des détails de résolution des problèmes sur les requêtes Redshift Spectrum au niveau du segment. Un segment peut exécuter une analyse de table externe. Cette vue est dérivée de la vue système SVL_S3LOG, mais n'affiche pas le niveau tranche pour les requêtes exécutées sur un cluster de mise à l'échelle de la simultanéité.

> **Note**
>
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires aux vues dotées du préfixe SVL, si ce n'est que les vues SVL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_S3LOG est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Pour plus d'informations sur SVL_S3LOG, consultez SVL_S3LOG.

Colonnes de la table

Nom de la colonne	Type de données	Description
pid	entier	ID du processus.
query	entier	ID de requête.
segment	entier	Numéro du segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes.
étape	entier	Étape de la requête exécutée.
node	entier	Numéro du nœud.
eventtime	timestamp	L'heure UTC à laquelle l'événement est enregistré.
message	char(512)	Message pour l'entrée de journal.

Exemple de requête

L'exemple suivant interroge SVCS_S3LOG par rapport à la dernière requête exécutée.

```
select *
```

```
from svcs_s3log
where query = pg_last_query_id()
order by query,segment;
```

SVCS_S3PARTITION_SUMMARY

Utilisez la vue SVCS_S3PARTITION_SUMMARY pour obtenir un résumé du traitement de la partition des requêtes Redshift Spectrum au niveau du segment. Un segment peut exécuter une analyse de table externe.

> **Note**
>
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires aux vues dotées du préfixe SVL, si ce n'est que les vues SVL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_S3PARTITION_SUMMARY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Pour plus d'informations sur SVL_S3PARTITION, consultez SVL_S3PARTITION.

Colonnes de la table

Nom de la colonne	Type de données	Description
query	entier	ID de requête. Vous pouvez utiliser cette valeur pour joindre d'autres vues et tables système.
segment	entier	Numéro du segment. Requête composée de plusieurs segments.
assignment	char(1)	Type d'attribution de la partition entre les nœuds.

Nom de la colonne	Type de données	Description
min_start time	timestamp	Heure UTC à laquelle le traitement de la partition a commencé.
max_endti me	timestamp	Heure UTC à laquelle le traitement de la partition s'est terminé.
min_durat ion	bigint	Durée minimum de traitement de la partition utilisée par un nœud pour cette requête (en microsecondes).
max_durat ion	bigint	Durée maximum de traitement de la partition utilisée par un nœud pour cette requête (en microsecondes).
avg_durat ion	bigint	Durée moyenne de traitement de la partition utilisée par un nœud pour cette requête (en microsecondes).
total_par titions	entier	Nombre total de partitions dans une table externe.
qualified _partitions	entier	Nombre total de partitions qualifiées.
min_assig ned_parti tions	entier	Nombre minimum de partitions attribuées sur un nœud.
max_assig ned_parti tions	entier	Nombre maximum de partitions attribuées sur un nœud.
avg_assig ned_parti tions	bigint	Nombre moyen de partitions attribuées sur un nœud.

Exemple de requête

L'exemple suivant permet d'obtenir les détails de l'analyse de la partition pour la dernière requête exécutée.

```
select query, segment, assignment, min_starttime, max_endtime, min_duration,
 avg_duration
from svcs_s3partition_summary
where query = pg_last_query_id()
order by query,segment;
```

SVCS_S3QUERY_SUMMARY

Utilisez la vue SVL_S3QUERY_SUMMARY pour obtenir un résumé de toutes les requêtes Redshift Spectrum (requêtes S3) exécutées sur le système. Un segment peut exécuter une analyse de table externe.

> **Note**
>
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires aux vues dotées du préfixe SVL, si ce n'est que les vues SVL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_S3QUERY_SUMMARY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Pour plus d'informations sur SVL_S3QUERY, consultez SVL_S3QUERY.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée donnée.

Nom de la colonne	Type de données	Description
query	entier	ID de requête. Vous pouvez utiliser cette valeur pour joindre d'autres vues et tables système.
xid	bigint	ID de transaction.
pid	entier	ID du processus.
segment	entier	Numéro du segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure UTC à laquelle la requête Redshift Spectrum de ce segment a commencé à s'exécuter. Un segment peut exécuter une analyse de table externe.
endtime	timestamp	Heure UTC à laquelle la requête Redshift Spectrum de ce segment s'est terminée. Un segment peut exécuter une analyse de table externe.
elapsed	entier	Durée nécessaire à l'exécution de la requête Redshift Spectrum dans ce segment (en microsecondes).
aborted	entier	Si une requête a été arrêtée par le système ou annulée par l'utilisateur, cette colonne contient **1**. Si la requête est terminée, cette colonne contient **0**.
external_table_name	char(136)	Format interne du nom externe de la table pour l'analyse de la table externe.
file_format	character(16)	Format de fichier des données de la table externe.
is_partitioned	char(1)	Si cette valeur de colonne est true (**t**), indique que la table externe est partitionnée.

Nom de la colonne	Type de données	Description
is_rrscan	char(1)	Si cette valeur de colonne est true (**t**), indique qu'une analyse à plage restreinte a été appliquée.
is_nested	varchar(1)	Si cette valeur de colonne est true (**t**), indique que le type de données de la colonne imbriquée est accessible.
s3_scanned_rows	bigint	Nombre de lignes analysées à partir d'Amazon S3 et envoyées à la couche Redshift Spectrum.
s3_scanned_bytes	bigint	Nombre d'octets analysés à partir d'Amazon S3 et envoyés à la couche Redshift Spectrum, en fonction des données compressées.
s3query_returned_rows	bigint	Nombre de lignes retournées par la couche Redshift Spectrum au cluster.
s3query_returned_bytes	bigint	Nombre d'octets retournés par la couche Redshift Spectrum au cluster. Si le volume des données renvoyées à Amazon Redshift est important, les performances du système peuvent être affectées.
fichiers	entier	Nombre de fichiers traités pour cette requête Redshift Spectrum. Les avantages du traitement parallèle sont amoindris s'il y a peu de fichiers.
files_max	entier	Nombre maximal de fichiers traités sur une tranche.
files_avg	entier	Nombre moyen de fichiers traités sur une tranche.
splits	bigint	Nombre de divisions traitées pour ce segment. Nombre de divisions traitées sur cette tranche. Avec des fichiers de données divisibles volumineux, par exemple, des fichiers de données supérieurs à environ 512 Mo, Redshift Spectrum essaie de diviser les fichiers en plusieurs demandes S3 de traitement parallèle.

Nom de la colonne	Type de données	Description
splits_max	entier	Nombre maximal de divisions traitées sur cette tranche.
splits_avg	bigint	Nombre moyen de divisions traitées sur cette tranche.
total_split_size	bigint	Taille totale de toutes les divisions traitées.
max_split_size	bigint	Taille de division maximale traitée, en octets.
avg_split_size	bigint	Taille de division moyenne traitée, en octets.
total_retries	bigint	Nombre total de nouvelles tentatives pour la requête Redshift Spectrum de ce segment.
max_retries	entier	Nombre maximal de nouvelles tentatives pour un fichier traité spécifique.
max_request_duration	bigint	Durée maximum d'une demande de fichier spécifique (en microsecondes). Les requêtes de longue durée peuvent indiquer la présence d’un goulot d’étranglement.
avg_request_duration	bigint	Durée moyenne des demandes de fichier (en microsecondes).
max_request_parallelism	entier	Nombre maximum de demandes parallèles sur une tranche pour cette requête Redshift Spectrum.
avg_request_parallelism	double precision	Nombre moyen de demandes parallèles sur une tranche pour cette requête Redshift Spectrum.

Nom de la colonne	Type de données	Description
total_slowdown_count	bigint	Nombre total de demandes Amazon S3 avec une erreur de ralentissement qui se produit lors de l'analyse de la table externe.
max_slowdown_count	entier	Nombre maximum de demandes Amazon S3 avec une erreur de ralentissement qui se produit lors de l'analyse de la table externe sur une tranche.

Exemple de requête

L'exemple suivant permet d'obtenir les détails de l'étape d'analyse pour la dernière requête exécutée.

```
select query, segment, elapsed, s3_scanned_rows, s3_scanned_bytes,
 s3query_returned_rows, s3query_returned_bytes, files
from svcs_s3query_summary
where query = pg_last_query_id()
order by query,segment;
```

```
query | segment | elapsed | s3_scanned_rows | s3_scanned_bytes | s3query_returned_rows
 | s3query_returned_bytes | files
------+---------+---------+-----------------+------------------+----------------------
+------------------------+------
 4587 |       2 |   67811 |               0 |                0 |                    0
 |                      0 |     0
 4587 |       2 |  591568 |          172462 |         11260097 |                 8513
 |                 170260 |     1
 4587 |       2 |  216849 |               0 |                0 |                    0
 |                      0 |     0
 4587 |       2 |  216671 |               0 |                0 |                    0
 |                      0 |     0
```

SVCS_STREAM_SEGS

Affiche les relations entre les flux et les segments simultanés.

> **Note**
>
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires aux tables dotées du préfixe STL, si ce n'est que les tables STL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_STREAM_SEGS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
stream	entier	Ensemble des segments simultanés d'une requête.
segment	entier	Numéro qui identifie le segment de requête.

Exemples de requêtes

Pour afficher la relation entre les flux et les segments simultanés de la requête la plus récente, tapez la requête suivante :

```
select *
from svcs_stream_segs
where query = pg_last_query_id();

 query | stream | segment
-------+--------+---------
    10 |      1 |       2
    10 |      0 |       0
```

```
    10 |      2 |       4
    10 |      1 |       3
    10 |      0 |       1
(5 rows)
```

SVCS_UNLOAD_LOG

Utilisez SVCS_UNLOAD_LOG pour obtenir des détails sur les opérations UNLOAD.

SVCS_UNLOAD_LOG enregistre une ligne pour chaque fichier créé par une instruction UNLOAD. Par exemple, si une instruction UNLOAD crée 12 fichiers, SVCS_UNLOAD_LOG contient 12 lignes correspondantes. Cette vue est dérivée de la table système STL_UNLOAD_LOG, mais n'affiche pas le niveau tranche pour les requêtes exécutées sur un cluster de mise à l'échelle de la simultanéité.

> **Note**
>
> Les vues système dotées du préfixe SVCS fournissent des détails à propos des requêtes sur le cluster principal et les clusters de mise à l'échelle de la simultanéité. Elles sont similaires aux tables dotées du préfixe STL, si ce n'est que les tables STL fournissent des informations uniquement pour les requêtes exécutées sur le cluster principal.

SVCS_UNLOAD_LOG est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête.
pid	entier	ID de processus associé à l'instruction de la requête.
path	character(1280)	Chemin d'objet Amazon S3 complet du fichier.
start_time	timestamp	Heure de début de l'opération UNLOAD.

Nom de la colonne	Type de données	Description
end_time	timestamp	Heure de fin de l'opération UNLOAD.
line_count	bigint	Nombre de lignes déchargées dans le fichier.
transfer_size	bigint	Nombre d'octets transférés.
file_format	character(10)	Nombre de fichiers déchargés

Exemple de requête

Pour obtenir la liste des fichiers écrits sur Amazon S3 par une commande UNLOAD, vous pouvez appeler une opération de liste Amazon S3 après la fin d'UNLOAD ; cependant, en fonction de la vitesse à laquelle vous émettez l'appel, la liste peut être incomplète parce qu'une opération Amazon S3 est cohérente à terme. Pour obtenir immédiatement une liste complète et faisant autorité, interrogez SVCS_UNLOAD_LOG.

La requête suivante renvoie le chemin d'accès pour les fichiers créés par une instruction UNLOAD pour la dernière requête exécutée :

```
select query, substring(path,0,40) as path
from svcs_unload_log
where query = pg_last_query_id()
order by path;
```

Cette commande renvoie l'exemple de sortie suivant :

```
 query |             path
 ------+-----------------------------------------
  2320 | s3://amzn-s3-demo-bucket/venue0000_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0001_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0002_part_00
  2320 | s3://amzn-s3-demo-bucket/venue0003_part_00
(4 rows)
```

Vues SVL pour le cluster principal

Les vues SVL sont des vues système dans Amazon Redshift qui contiennent des références aux tables STL et aux journaux pour plus d'informations.

Ces vues offrent un accès plus rapide et plus facile aux données fréquemment interrogées trouvées dans ces tables.

> **Note**
>
> La vue SVL_QUERY_SUMMARY contient uniquement des informations sur les requêtes exécutées par Amazon Redshift, et non sur d'autres commandes DDL ou d'utilitaire. Pour obtenir une liste complète et des informations sur toutes les instructions exécutées par Amazon Redshift, y compris les commandes DDL et utilitaires, vous pouvez interroger la vue SVL_STATEMENTTEXT.

Rubriques

- SVL_AUTO_WORKER_ACTION
- SVL_COMPILE
- SVL_DATASHARE_CHANGE_LOG
- SVL_DATASHARE_CROSS_REGION_USAGE
- SVL_DATASHARE_USAGE_CONSUMER
- SVL_DATASHARE_USAGE_PRODUCER
- SVL_FEDERATED_QUERY
- SVL_MULTI_STATEMENT_VIOLATIONS
- SVL_MV_REFRESH_STATUS
- SVL_QERROR
- SVL_QLOG
- SVL_QUERY_METRICS
- SVL_QUERY_METRICS_SUMMARY
- SVL_QUERY_QUEUE_INFO
- SVL_QUERY_REPORT

- SVL_QUERY_SUMMARY
- SVL_RESTORE_ALTER_TABLE_PROGRESS
- SVL_S3LIST
- SVL_S3LOG
- SVL_S3PARTITION
- SVL_S3PARTITION_SUMMARY
- SVL_S3QUERY
- SVL_S3QUERY_SUMMARY
- SVL_S3RETRIES
- SVL_SPATIAL_SIMPLIFICATION
- SVL_SPECTRUM_SCAN_ERROR
- SVL_STATEMENTTEXT
- SVL_STORED_PROC_CALL
- SVL_STORED_PROC_MESSAGES
- SVL_TERMINATE
- SVL_UDF_LOG
- SVL_USER_INFO
- SVL_VACUUM_PERCENTAGE

SVL_AUTO_WORKER_ACTION

Enregistre les actions automatisées d'Amazon Redshift sur les tables définies pour l'optimisation automatique.

SVL_AUTO_WORKER_ACTION est visible par tous les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
table_id	entier	L'identificateur de table.

Nom de la colonne	Type de données	Description
type	character (32)	Le type de recommandation. Les valeurs possibles sont distkey et sortkey.
status	character (128)	Le statut d'achèvement de la recommandation. Les valeurs possibles sont Start, Complete, Skipped, Abort, Checkpoint et Failed.
eventtime	timestamp	Horodatage de la colonne `status`.
sequence	entier	Le numéro de séquence d'une valeur `previous_state` tronquée. Lorsqu'une seule instruction `previous_state` contient plus de 200 caractères, des lignes supplémentaires sont enregistrées pour cette valeur. Sequence 0 correspond à la première ligne, 1 à la deuxième, et ainsi de suite.
previous_ state	character (200)	Le style de distribution précédent et les clés de tri de la table avant d'appliquer la recommandation. La valeur est tronquée en incréments de 200 caractères.

Voici quelques exemples de valeurs de la colonne `status` :

- Skipped:Table not found.
- Skipped:Recommendation is empty.
- Skipped:Apply sortkey recommendation is disabled.
- Skipped:Retry exceeds the maximum limit for a table.
- Skipped:Table column has changed.
- Abort:This table is not AUTO.
- Abort:This table has been recently converted.
- Abort:This table exceeds table size threshold.
- Abort:This table is already the recommended style.
- Point de contrôle : *21.9963* % de progression.

Exemples de requêtes

Dans l'exemple suivant, les lignes de résultats affichent les actions d'Amazon Redshift.

```
select table_id, type, status, eventtime, sequence, previous_state
from SVL_AUTO_WORKER_ACTION;
```

```
 table_id |  type   |                        status                        |
 eventtime          | sequence | previous_state
----------+---------+------------------------------------------------------
+----------------------------+----------+----------------
   118082 | sortkey | Start                                                | 2020-08-22
 19:42:20.727049 | 0        |
   118078 | sortkey | Start                                                | 2020-08-22
 19:43:54.728819 | 0        |
   118082 | sortkey | Start                                                | 2020-08-22
 19:42:52.690264 | 0        |
   118072 | sortkey | Start                                                | 2020-08-22
 19:44:14.793572 | 0        |
   118082 | sortkey | Failed                                               | 2020-08-22
 19:42:20.728917 | 0        |
   118078 | sortkey | Complete                                             | 2020-08-22
 19:43:54.792705 | 0        | SORTKEY: None;
   118086 | sortkey | Complete                                             | 2020-08-22
 19:42:00.72635  | 0        | SORTKEY: None;
   118082 | sortkey | Complete                                             | 2020-08-22
 19:43:34.728144 | 0        | SORTKEY: None;
   118072 | sortkey | Skipped:Retry exceeds the maximum limit for a table. | 2020-08-22
 19:44:46.706155 | 0        |
   118086 | sortkey | Start                                                | 2020-08-22
 19:42:00.685255 | 0        |
   118082 | sortkey | Start                                                | 2020-08-22
 19:43:34.69531  | 0        |
   118072 | sortkey | Start                                                | 2020-08-22
 19:44:46.703331 | 0        |
   118082 | sortkey | Checkpoint: progress 14.755079%                      | 2020-08-22
 19:42:52.692828 | 0        |
   118072 | sortkey | Failed                                               | 2020-08-22
 19:44:14.796071 | 0        |
   116723 | sortkey | Abort:This table is not AUTO.                        | 2020-10-28
 05:12:58.479233 | 0        |
```

```
  110203 | distkey | Abort:This table is not AUTO.                    | 2020-10-28
05:45:54.67259  | 0       |
```

SVL_COMPILE

Les enregistrements compilent l'heure et l'emplacement de chaque segment de requête des requêtes.

SVL_COMPILE est visible de tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> SVL_COMPILE contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_QUERY_HISTORY Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Pour plus d'informations sur SVCS_COMPILE, consultez SVCS_COMPILE.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
xid	bigint	ID de transaction associé à l'instruction.
pid	entier	ID de processus associé à l'instruction.
query	entier	ID de requête. Permet de joindre d'autres tables système et vues.
segment	entier	Segment de la requête à compiler.

Nom de la colonne	Type de données	Description
locus	entier	Emplacement où le segment s'exécute. **1** s'il s'exécute sur un nœud de calcul et **2** s'il s'exécute sur le nœud principal.
starttime	timestamp	Heure UTC à laquelle la compilation a commencé.
endtime	timestamp	Heure UTC à laquelle la compilation a pris fin.
compile	entier	**0** si la compilation a été réutilisée, **1** si le segment a été compilé.

Exemples de requêtes

Dans cet exemple, les requêtes 35878 et 35879 ont exécuté la même instruction SQL. La colonne de compilation de la requête 35878 affiche 1 pour quatre segments de requête, ce qui indique que les segments ont été compilés. La requête 35879 affiche `0` dans la colonne de compilation de chaque segment, ce qui indique que les segments n'avaient pas besoin d'être compilés à nouveau.

```
select userid, xid,  pid, query, segment, locus,
datediff(ms, starttime, endtime) as duration, compile
from svl_compile
where query = 35878 or query = 35879
order by query, segment;

 userid |  xid   |  pid  | query | segment | locus | duration | compile
--------+--------+-------+-------+---------+-------+----------+---------
    100 | 112780 | 23028 | 35878 |       0 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       1 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       2 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       3 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       4 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       5 |     1 |        0 |       0
    100 | 112780 | 23028 | 35878 |       6 |     1 |     1380 |       1
    100 | 112780 | 23028 | 35878 |       7 |     1 |     1085 |       1
    100 | 112780 | 23028 | 35878 |       8 |     1 |     1197 |       1
    100 | 112780 | 23028 | 35878 |       9 |     2 |      905 |       1
    100 | 112782 | 23028 | 35879 |       0 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       1 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       2 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       3 |     1 |        0 |       0
```

```
    100 | 112782 | 23028 | 35879 |       4 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       5 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       6 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       7 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       8 |     1 |        0 |       0
    100 | 112782 | 23028 | 35879 |       9 |     2 |        0 |       0
(20 rows)
```

SVL_DATASHARE_CHANGE_LOG

Enregistre la vue consolidée pour le suivi des modifications apportées aux unités de partage des données sur les clusters producteur et consommateur.

SVL_DATASHARE_CHANGE_LOG est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_DATASHARE_CHANGE_LOG. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	L'ID de l'utilisateur qui effectue l'action.
nom d'utilisateur	varchar(128)	Le nom de l'utilisateur qui effectue l'action.
pid	entier	ID du processus.
xid	bigint	ID de la transaction.
share_id	entier	L'ID de l'unité de partage des données affecté.
share_name	varchar(128)	Le nom du datashare.

Nom de la colonne	Type de données	Description
source_database_id	entier	L'ID de la base de données à laquelle appartient l'unité de partage des données.
source_database_name	varchar(128)	Le nom de la base de données à laquelle appartient le datashare.
consumer_database_id	entier	L'ID de la base de données importée depuis l'unité de partage des données.
consumer_database_name	varchar(128)	Le nom de la base de données importée depuis le datashare.
arn	varchar(192)	L'ARN de la ressource qui soutient la base de données importée.
recordtime	timestamp	Horodatage de l'action.
action	varchar(128)	L'action en cours d'exécution. Les valeurs possibles sont CREATE DATASHARE, DROP DATASHARE, GRANT ALTER, REVOKE ALTER, GRANT SHARE, REVOKE SHARE, ALTER ADD, ALTER REMOVE, ALTER SET, GRANT USAGE, REVOKE USAGE, CREATE DATABASE, GRANT ou REVOKE USAGE sur une base de données partagée, DROP SHARED DATABASE, ALTER SHARED DATABASE.
status	entier	Statut de l'action. Les valeurs possibles sont SUCCESS et ERROR-ERROR CODE.
share_object_type	varchar(64)	Le type d'objet de base de données qui a été ajouté ou supprimé de l'unité de partage des données. Les valeurs possibles sont les schémas, les tables, les colonnes, les fonctions et les vues. Il s'agit d'un champ pour le cluster producteur.

Nom de la colonne	Type de données	Description
share_object_id	entier	L'ID de l'objet de base de données qui a été ajouté ou supprimé de l'unité de partage des données. Il s'agit d'un champ pour le cluster producteur.
share_object_name	varchar(128)	Le nom de l'objet de base de données qui a été ajouté ou supprimé de l'unité de partage des données. Il s'agit d'un champ pour le cluster producteur.
target_user_type	varchar(16)	Le type d'utilisateurs ou de groupes auxquels un privilège a été accordé. Il s'agit d'un domaine à la fois pour le cluster producteur et consommateur.
target_userid	entier	L'ID des utilisateurs ou des groupes auxquels un privilège a été accordé. Il s'agit d'un domaine à la fois pour le cluster producteur et consommateur.
target_username	varchar(128)	Le nom des utilisateurs ou des groupes auxquels un privilège a été accordé. Il s'agit d'un domaine à la fois pour le cluster producteur et consommateur.
consumer_account	varchar(16)	L'ID de compte du consommateur de données. Il s'agit d'un champ pour le cluster producteur.
consumer_namespace	varchar(64)	L'espace de noms du compte consommateur de données. Il s'agit d'un champ pour le cluster producteur.
producer_account	varchar(16)	L'ID du compte producteur auquel appartient l'unité de partage des données. Il s'agit d'un champ pour le cluster consommateur.
producer_namespace	varchar(64)	L'espace de noms du compte producteur auquel appartient l'unité de partage des données. Il s'agit d'un champ pour le cluster consommateur.
attribute_name	varchar(64)	Le nom d'un attribut de l'unité de partage des données ou de la base de données partagée.

Nom de la colonne	Type de données	Description
attribute_value	varchar(128)	La valeur d'un attribut de l'unité de partage des données ou de la base de données partagée.
message	varchar(512)	Le message d'erreur lorsqu'une action échoue.

Exemples de requêtes

L'exemple suivant montre une vue SVL_DATASHARE_CHANGE_LOG.

```
SELECT DISTINCT action
FROM svl_datashare_change_log
WHERE share_object_name LIKE 'tickit%';

         action
 ----------------------
  "ALTER DATASHARE ADD"
```

SVL_DATASHARE_CROSS_REGION_USAGE

Utilisez la vue SVL_DATASHARE_CROSS_REGION_USAGE pour obtenir un résumé de l'utilisation des données transférées d'une région à l'autre, causée par une requête d'unité de partage des données entre les régions. SVL_DATASHARE_CROSS_REGION_USAGE agrège les détails au niveau du segment.

SVL_DATASHARE_CROSS_REGION_USAGE est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_DATASHARE_CROSS_REGION_USAGE. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
query	entier	ID de la requête. Utilisez cette valeur pour joindre d'autres tables et vues du système.
segment	bigint	Numéro du segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes.
start_time	time	Heure UTC à laquelle le transfert de données a commencé.
end_time	time	Heure UTC à laquelle le transfert de données s'est terminé.
transferred_data	bigint	Nombre d'octets de données transférés d'une région productrice à une région consommatrice.
source_region	char(255)	La région productrice à partir de laquelle la requête a transféré des données.
recordtime	timestamp	L'heure à laquelle l'action est enregistrée.

Exemples de requêtes

L'exemple suivant montre une vue SVL_DATASHARE_CROSS_REGION_USAGE.

```
SELECT query, segment, transferred_data, source_region
from svl_datashare_cross_region_usage
where query = pg_last_query_id()
order by query,segment;

  query | segment | transferred_data | source_region
--------+---------+------------------+---------------
 200048 |       2 |          4194304 |    us-west-1
 200048 |       2 |          4194304 |    us-east-2
```

SVL_DATASHARE_USAGE_CONSUMER

Enregistre l'activité et l'utilisation des datashares. Cette vue n’est pertinente que pour le cluster consommateur.

SVL_DATASHARE_USAGE_CONSOMMY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_DATASHARE_USAGE_CONSUMER. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d’utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	L’ID de l’utilisateur à l’origine de la requête.
pid	entier	ID du processus de leader exécutant la requête.
xid	bigint	Le contexte de la transaction actuelle.
request_id	varchar(50)	L’ID unique de l’appel d’API demandé.
request_type	varchar(25)	Le type de la requête envoyée au cluster producteur.
transaction_uid	varchar(50)	ID unique de la transaction.
recordtime	timestamp	L’heure à laquelle l’action est enregistrée.
status	entier	Le statut de l’appel d’API demandé.

Nom de la colonne	Type de données	Description
error	varchar(512)	Un message d'erreur.

Exemples de requêtes

L'exemple suivant montre une vue SVL_DATASHARE_USAGE_CONSUMER.

```
SELECT request_type, status, trim(error) AS error
FROM svl_datashare_usage_consumer

  request_type  | status | error
----------------+--------+--------
 "GET RELATION" |   0    |
```

SVL_DATASHARE_USAGE_PRODUCER

Enregistre l'activité et l'utilisation des datashares. Cette vue n'est pertinente que sur le cluster producteur.

SVL_DATASHARE_USAGE_PRODUCER est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_DATASHARE_USAGE_PRODUCER. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
share_id	entier	L'ID d'objet (OID) de l'unité de partage des données.

Nom de la colonne	Type de données	Description
share_name	varchar(128)	Le nom du datashare.
request_id	varchar(50)	L'ID unique de l'appel d'API demandé.
request_type	varchar(25)	Le type de la requête envoyée au cluster producteur.
object_type	varchar(64)	Le type de l'objet partagé à partir de l'unité de partage des données. Les valeurs possibles sont les schémas, les tables, les colonnes, les fonctions et les vues.
object_oid	entier	L'ID de l'objet partagé à partir de l'unité de partage des données.
nom d'objet	varchar(128)	Le nom de l'objet partagé à partir de l'unité de partage des données.
consumer_account	varchar(16)	Le compte consommateur relié au datashare pour le partage.
consumer_namespace	varchar(64)	L'espace de noms du compte consommateur relié à l'unité de partage des données pour le partage.
consumer_transaction_uid	varchar(50)	L'ID de transaction unique de l'instruction sur le cluster consommateur.
recordtime	timestamp	L'heure à laquelle l'action est enregistrée.
status	entier	Le statut du datashare.
error	varchar(512)	Un message d'erreur.
consumer_region	char(64)	Région dans laquelle est situé le cluster consommateur.

Exemples de requêtes

L'exemple suivant montre une vue SVL_DATASHARE_USAGE_PRODUCER.

```
SELECT DISTINCT request_type
FROM svl_datashare_usage_producer
WHERE object_name LIKE 'tickit%';

   request_type
 ------------------
   "GET RELATION"
```

SVL_FEDERATED_QUERY

Utilisez la vue SVL_FEDERATED_QUERY pour afficher des informations sur un appel de requête fédérée.

SVL_FEDERATED_QUERY est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_EXTERNAL_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur exécutant la requête.
xid	bigint	ID de transaction.
pid	entier	ID du processus de leader exécutant la requête.
query	entier	ID de requête d'un appel fédéré.

Nom de la colonne	Type de données	Description
sourcetype	character(32)	Type de source d'appel fédérée, par exemple, "PG".
recordtime	timestamp	Heure à laquelle une requête est envoyée pour fédération. UTC est utilisé.
querytext	character(4000)	Chaîne de requête envoyée au moteur PostgreSQL distant pour exécution.
num_rows	bigint	Le nombre de lignes renvoyées par la requête fédérée.
num_bytes	bigint	Le nombre d'octets renvoyés par la requête fédérée.
duration	bigint	Le temps (microsecondes) passé à récupérer des lignes à partir d'appels de curseurs. Il s'agit du temps passé à exécuter la requête fédérée, ainsi qu'à obtenir des résultats en retour.

Exemples de requêtes

Pour afficher des informations sur les appels de requête fédérés, exécutez la requête suivante.

```
select query, trim(sourcetype) as type, recordtime, trim(querytext) as "PG Subquery"
 from svl_federated_query where query = 4292;

 query | type |         recordtime         |                          pg subquery
-------+------+----------------------------
+------------------------------------------------------------
```

```
  4292 | PG   | 2020-03-27 04:29:58.485126 | SELECT "level" FROM functional.employees
 WHERE ("level" >= 6)
(1 row)
```

SVL_MULTI_STATEMENT_VIOLATIONS

Utilisez la vue SVL_MULTI_STATEMENT_VIOLATIONS pour obtenir un enregistrement complet de toutes les commandes SQL exécutées sur le système qui enfreignent les restrictions de bloc de transaction.

Des violations se produisent lorsque vous exécutez l'une des commandes SQL suivantes qu'Amazon Redshift restreint à l'intérieur d'un bloc de transaction ou de requêtes multi-instructions :

- CREATE DATABASE
- DROP DATABASE
- ALTER TABLE APPEND
- CREATE EXTERNAL TABLE
- DROP EXTERNAL TABLE
- RENAME EXTERNAL TABLE
- ALTER EXTERNAL TABLE
- CREATE TABLESPACE
- DROP TABLESPACE
- CREATE LIBRARY
- DROP LIBRARY
- REBUILDCAT
- INDEXCAT
- REINDEX DATABASE
- VACUUM
- GRANT
- COPY

> **Note**
>
> S'il y a des entrées dans cette vue, modifiez les applications et scripts SQL correspondants. Nous vous recommandons de modifier le code de votre application pour déplacer l'utilisation

de ces commandes SQL restreintes en dehors du bloc de transactions. Si vous avez besoin d'une assistance supplémentaire, contactez AWS le Support.

SVL_MULTI_STATEMENT_VIOLATIONS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	L'ID de l'utilisateur à l'origine de la violation.
database	character(32)	Nom de la base de données à laquelle l'utilisateur était connecté.
cmdname	character(20)	Le nom de la commande qui ne peut pas s'exécuter à l'intérieur d'un bloc de transaction ou d'une requête multi-instructions. Par exemple : CREATE DATABASE, DROP DATABASE, ALTER TABLE APPEND, CREATE EXTERNAL TABLE, DROP EXTERNAL TABLE, RENAME EXTERNAL TABLE, ALTER EXTERNAL TABLE, CREATE LIBRARY, DROP LIBRARY, REBUILDCAT, INDEXCAT, REINDEX DATABASE, VACUUM, GRANT sur des ressources externes, CLUSTER, COPY, CREATE TABLESPACE et DROP TABLESPACE.
xid	bigint	ID de transaction associé à l'instruction.
pid	entier	ID de processus de l'instruction.

Nom de la colonne	Type de données	Description
étiquette	caractère (320)	Le nom du fichier utilisé pour exécuter la requête ou une étiquette définie avec une commande SET QUERY_GRO UP. Si la requête n'est pas basée sur un fichier ou si le paramètre QUERY_GROUP n'est pas défini, le champ est vide.
starttime	timestamp	Heure exacte à laquelle l'exécution de l'instruction a démarré, avec six chiffres de précision pour les fractions de secondes, par exemple : **2009-06-12 11:29:19. 131358**
endtime	timestamp	Heure exacte à laquelle l'exécution de l'instruction s'est terminée, avec six chiffres de précision pour les fractions de secondes, par exemple : **2009-06-12 11:29:19. 193640**
sequence	entier	Lorsqu'une seule instruction contient plus de 200 caractère s, des lignes supplémentaires sont enregistrées pour l'instruction. Sequence 0 correspond à la première ligne, 1 à la deuxième, et ainsi de suite.
type	varchar(10)	Type d'instruction SQL : **QUERY**, **DDL** ou **UTILITY**.
text	character(200)	Texte SQL, par incréments de 200 caractères. Ce champ peut contenir des caractères spéciaux tels qu'une barre oblique inverse (\\) et un caractère de saut de ligne (\n).

Exemple de requête

La requête suivante renvoie plusieurs instructions comportant des violations.

```
select * from svl_multi_statement_violations order by starttime asc;

userid | database | cmdname | xid | pid | label | starttime | endtime | sequence | type
 | text
================================================================================================
```

```
1 | dev | CREATE DATABASE | 1034 | 5729 |label1 | ********* | ******* | 0 | DDL |
 create table c(b int);
1 | dev | CREATE DATABASE | 1034 | 5729 |label1 | ********* | ******* | 0 | UTILITY |
 create database b;
1 | dev | CREATE DATABASE | 1034 | 5729 |label1 | ********* | ******* | 0 | UTILITY |
 COMMIT
...
```

SVL_MV_REFRESH_STATUS

La vue SVL_MV_REFRESH_STATUS contient une ligne pour l'activité d'actualisation des vues matérialisées.

Pour plus d'informations sur les vues matérialisées, consultez Vues matérialisées dans Amazon Redshift.

SVL_MV_REFRESH_STATUS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_MV_REFRESH_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
db_name	char(128)	Base de données contenant la vue matérialisée.
userid	bigint	ID de l'utilisateur ayant effectué l'actualisation.
nom_schéma	char(128)	Schéma de la vue matérialisée.
mv_name	char(128)	Nom de la vue matérialisée.
xid	bigint	ID de transaction de l'actualisation.

Nom de la colonne	Type de données	Description
starttime	timestamp	Heure de début de l’actualisation.
endtime	timestamp	Heure de fin de l’actualisation.

Nom de la colonne	Type de données	Description
État	text	Statut de l'actualisation. Exemples de valeur possible : • Refresh successfully updated MV incrementally (L'actualisation a mis à jour la vue matérialisée de manière incrémentielle) S'il s'agit d'une vue matérialisée destinée à la diffusion en continu, le message peut comporter des qualificatifs supplémentaires concernant le nombre d'enregistrements. Tel est le cas des éléments suivants : • Stream returned no new data (Le flux n'a pas renvoyé de données) – Aucun enregistrement n'a été récupéré. • Tous les enregistrements provenant du flux ont été ignorés : des enregistrements ont été récupérés, mais tous ont été ignorés en raison d'une erreur. • Certains enregistrements du flux ont été ignorés : des enregistrements ont été récupérés, mais certains ont été ignorés en raison d'une erreur. S'il n'y a pas de qualificateurs, cela indique qu'au moins un enregistrement a été récupéré et que tous les enregistrements sont disponibl es dans la vue matérialisée. Il reste un qualifica teur possible : • The stream may contain more data (Il se peut que le flux contienne plus de données) – L'actualisation s'est terminée avant qu'Amazon Redshift ait déterminé qu'il n'y

Nom de la colonne	Type de données	Description
		avait plus d'enregistrements à consommer. Le flux est peut-être à jour, mais cela n'a pas été confirmé par Amazon Redshift. • Refresh successfully recomputed MV from scratch (L'actualisation a recalculé correctem ent la vue matérialisée) • Refresh partially updated MV incrementally up to an active transaction (L'actualisation a mis partiellement à jour la vue matérialisée de manière incrémentielle jusqu'à une transaction active) • MV was already updated (La vue matérialisée a déjà été mise à jour) • Refresh failed. A base table column was renamed (L'actualisation a échoué. Une colonne de table de base a été renommée) • Refresh failed. A base table column type was changed (L'actualisation a échoué. Un type de colonne de table de base a été modifié) • Refresh failed. A base table was renamed (L'actualisation a échoué. Une table de base a été renommée) • Refresh failed due to an internal error (L'actual isation a échoué en raison d'une erreur interne) • Refresh failed. A base table column was dropped (L'actualisation a échoué. Une colonne de table de base a été supprimée) • Refresh failed. Schema of MV was renamed (L'actualisation a échoué. Le schéma de la vue matérialisée a été renommé)

Nom de la colonne	Type de données	Description
		• Refresh failed. MV was not found (L'actualisation a échoué. La vue matérialisée est introuvable) • Auto refresh aborted due to excessive user workload (Actualisation automatique abandonnée en raison d'une charge de travail utilisateur excessive) • Refresh failed. Serializable isolation violation (L'actualisation a échoué. Violation d'isolement sérialisable)
refresh_type	char(32)	La définition du type d'actualisation. Les valeurs sont les suivantes : Manuel et Auto.

Exemple de requête

Pour afficher le statut d'actualisation des vues matérialisées, exécutez la requête suivante.

```
select * from svl_mv_refresh_status;
```

Cette requête renvoie l'exemple de sortie suivant :

```
 db_name | userid |  schema   |  name   |  xid  |         starttime          |
       endtime           |                        status                        |
 refresh_type
---------+--------+-----------+---------+-------+----------------------------
+----------------------------+------------------------------------------------------
+-----------------
 dev     |    169 | mv_schema | mv_test |  6640 | 2020-02-14 02:26:53.497935 |
 2020-02-14 02:26:53.556156 | Refresh successfully recomputed MV from scratch     |
 Manual
 dev     |    166 | mv_schema | mv_test |  6517 | 2020-02-14 02:26:39.287438 |
 2020-02-14 02:26:39.349539 | Refresh successfully updated MV incrementally       |
 Auto
```

```
dev     |    162 | mv_schema | mv_test |  6388 | 2020-02-14 02:26:27.863426 |
2020-02-14 02:26:27.918307 | Refresh successfully recomputed MV from scratch     |
Manual
dev     |    161 | mv_schema | mv_test |  6323 | 2020-02-14 02:26:20.020717 |
2020-02-14 02:26:20.080002 | Refresh successfully updated MV incrementally       |
Auto
dev     |    161 | mv_schema | mv_test |  6301 | 2020-02-14 02:26:05.796146 |
2020-02-14 02:26:07.853986 | Refresh successfully recomputed MV from scratch     |
Manual
dev     |    153 | mv_schema | mv_test |  6024 | 2020-02-14 02:25:18.762335 |
2020-02-14 02:25:20.043462 | MV was already updated                              |
Manual
dev     |    143 | mv_schema | mv_test |  5557 | 2020-02-14 02:24:23.100601 |
2020-02-14 02:24:23.100633 | MV was already updated                              |
Manual
dev     |    141 | mv_schema | mv_test |  5447 | 2020-02-14 02:23:54.102837 |
2020-02-14 02:24:00.310166 | Refresh successfully updated MV incrementally       |
Auto
dev     |      1 | mv_schema | mv_test |  5329 | 2020-02-14 02:22:26.328481 |
2020-02-14 02:22:28.369217 | Refresh successfully recomputed MV from scratch     |
Auto
dev     |    138 | mv_schema | mv_test |  5290 | 2020-02-14 02:21:56.885093 |
2020-02-14 02:21:56.885098 | Refresh failed. MV was not found                    |
Manual
```

SVL_QERROR

La vue SVL_QERROR est obsolète.

SVL_QLOG

La vue SVL_QLOG contient un journal de toutes les requêtes exécutées sur la base de données.

Amazon Redshift crée la vue SVL_QLOG comme sous-ensemble lisible d'informations de la table STL_QUERY. Utilisez ce tableau pour rechercher l'ID de requête d'une requête récemment exécutée ou pour voir combien de temps nécessite l'exécution d'une requête.

SVL_QLOG est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. Vous pouvez utiliser cet ID pour joindre d'autres vues et tables système.
xid	bigint	ID de transaction.
pid	entier	ID de processus associé à la requête.
starttime	timestamp	Heure exacte à laquelle l'exécution de l'instruction a démarré, avec six chiffres de précision pour les fractions de secondes, par exemple : **`2009-06-12 11:29:19.131358`**
endtime	timestamp	Heure exacte à laquelle l'exécution de l'instruction s'est terminée, avec six chiffres de précision pour les fractions de secondes, par exemple : **`2009-06-12 11:29:19.193640`**
elapsed	bigint	Durée nécessaire à l'exécution de la requête (en microsecondes).
aborted	entier	Si une requête a été arrêtée par le système ou annulée par l'utilisateur, cette colonne contient **`1`**. Si la requête est terminée, cette colonne contient **`0`**. Les requêtes qui sont annulées à des fins de gestion de la charge de travail (et redémarrées par la suite) ont aussi la valeur **`1`** dans cette colonne.

Nom de la colonne	Type de données	Description
étiquette	caractère (320)	Nom du fichier utilisé pour exécuter la requête ou étiquette définie avec une commande SET QUERY_GROUP. Si la requête n'est pas basée sur un fichier ou si le paramètre QUERY_GROUP n'est pas défini, la valeur du champ est `default`.
substring	character(60)	Texte de la requête tronqué.
source_query	entier	Si la requête utilisait la mise en cache des résultats, ID de la requête à la source des résultats mis en cache. Si la mise en cache des résultats n'a pas été utilisée, cette valeur de champ est `NULL`.
concurrency_scaling_status_txt	text	Description indiquant si la requête a été exécutée sur le cluster principal ou sur un cluster de mise à l'échelle de simultanéité.
from_sp_call	entier	ID de requête de l'appel de procédure, si la requête a été appelée à partir d'une procédure stockée. Si la requête n'a pas été exécutée dans le cadre d'une procédure stockée, ce champ a pour valeur `NULL`.

Exemples de requêtes

L'exemple suivant renvoie l'ID de requête, l'heure d'exécution et le texte de la requête tronqué pour les cinq requêtes de base de données les plus récentes exécutées par l'utilisateur avec `userid = 100`.

```
select query, pid, elapsed, substring from svl_qlog
where userid = 100
order by starttime desc
limit 5;

 query  |  pid  | elapsed  |           substring
--------+-------+----------+-----------------------------------------------
```

```
 187752 | 18921 | 18465685 | select query, elapsed, substring from svl_...
 204168 |  5117 |    59603 | insert into testtable values (100);
 187561 | 17046 |  1003052 | select * from pg_table_def where tablename...
 187549 | 17046 |  1108584 | select * from STV_WLM_SERVICE_CLASS_CONFIG
 187468 | 17046 |  5670661 | select * from pg_table_def where schemaname...
(5 rows)
```

L'exemple suivant renvoie le nom du script SQL (colonne LABEL) et le temps écoulé pour une requête qui a été annulée (**`aborted=1`**) :

```
select query, elapsed, trim(label) querylabel
from svl_qlog where aborted=1;

 query | elapsed  |       querylabel
-------+----------+-------------------------
    16 |  6935292 | alltickittablesjoin.sql
(1 row)
```

SVL_QUERY_METRICS

La vue SVL_QUERY_METRICS affiche les métriques des requêtes terminées. Cette vue dérive de la table système STL_QUERY_METRICS. Utilisez les valeurs de cette vue afin de vous aider à déterminer les seuils permettant de définir les règles de surveillance des requêtes. Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.

SVL_QUERY_METRICS est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a exécuté la requête qui a généré l'entrée.
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
service_class	entier	ID de la file d'attente de requêtes WLM (classe de service). Les files d'attente de requêtes sont définies dans la configura tion WLM. Les métriques sont communiquées uniquement pour les files d'attente définies par l'utilisateur. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.
dimension	varchar(24)	Dimension à laquelle la métrique est présentée. Les valeurs possibles sont query, segment et step.
segment	entier	Numéro de segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Les segments de requête peuvent s'exécuter en parallèle . Chaque segment s'exécute dans un processus unique. Si la valeur du segment est 0, les valeurs du segment de métriques sont reportées au niveau de la requête.
étape	entier	ID du type de l'étape exécutée. La description du type d'étape est indiquée dans la colonne `step_label` .
step_label	varchar(30)	Type d'étape exécutée.
query_cpu_time	bigint	Temps UC utilisé par la requête (en secondes). Le temps UC diffère de la durée d'exécution de la requête.
query_blo cks_read	bigint	Nombre de blocs d'1 Mo lus par la requête.

Nom de la colonne	Type de données	Description
query_execution_time	bigint	Délai écoulé pour l'exécution d'une requête (en secondes). Le délai d'exécution n'inclut pas le temps d'attente dans une file d'attente. Voir `query_queue_time` pour le temps de mise en file d'attente.
query_cpu_usage_percent	bigint	Pourcentage de la capacité de l'UC utilisée par la requête.
query_temp_blocks_to_disk	bigint	Quantité d'espace disque utilisé par une requête pour écrire des résultats intermédiaires, en Mo.
segment_execution_time	bigint	Délai écoulé pour l'exécution d'un seul segment (en secondes).
cpu_skew	numeric(38,2)	Ratio de l'utilisation maximale de l'UC pour une tranche afin d'obtenir l'utilisation moyenne de l'UC pour toutes les tranches. Cette métrique est définie au niveau du segment.
io_skew	numeric(38,2)	Ratio du nombre maximal de blocs lus (I/O) pour une tranche quelconque afin d'obtenir le nombre moyen de blocs lus pour toutes les tranches.
scan_row_count	bigint	Nombre de lignes dans une étape d'analyse. Le nombre de lignes correspond au nombre total de lignes émises avant le filtrage des lignes marquées pour la suppression (lignes fantôme) et avant l'application des filtres de requête définis par l'utilisateur.
join_row_count	bigint	Nombre de lignes traitées dans une étape de jonction.
nested_loop_join_row_count	bigint	Nombre de lignes dans une jonction de boucles imbriquées.
return_row_count	bigint	Nombre de lignes retournées par la requête.

Nom de la colonne	Type de données	Description
spectrum_scan_row_count	bigint	Le nombre de lignes analysées par Amazon Redshift Spectrum dans Amazon S3.
spectrum_scan_size_mb	bigint	Quantité de données, en Mo, analysées par Amazon Redshift Spectrum dans Amazon S3.
query_queue_time	bigint	Durée, en secondes, pendant laquelle la requête a été mise en file d'attente.

SVL_QUERY_METRICS_SUMMARY

La vue SVL_QUERY_METRICS_SUMMARY affiche les valeurs maximales des métriques des requêtes terminées. Cette vue dérive de la table système STL_QUERY_METRICS. Utilisez les valeurs de cette vue afin de vous aider à déterminer les seuils permettant de définir les règles de surveillance des requêtes. Pour plus d'informations sur les règles et métriques pour la surveillance des requêtes pour Amazon Redshift, consultez Règles de surveillance de requête WLM.

SVL_QUERY_METRICS_SUMMARY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a exécuté la requête qui a généré l'entrée.

Nom de la colonne	Type de données	Description
query	entier	ID de requête. La colonne de requête peut servir à joindre les autres tables système et les vues.
service_class	entier	ID de la file d'attente de requêtes WLM (classe de service). Les files d'attente de requêtes sont définies dans la configura tion WLM. Les métriques sont communiquées uniquement pour les files d'attente définies par l'utilisateur. Pour obtenir la liste des classes de service IDs, consultezClasse de service WLM IDs.
query_cpu_time	bigint	Temps UC utilisé par la requête (en secondes). Le temps UC diffère de la durée d'exécution de la requête.
query_blo cks_read	bigint	Nombre de blocs d'1 Mo lus par la requête.
query_exe cution_time	bigint	Délai écoulé pour l'exécution d'une requête (en secondes). Le délai d'exécution n'inclut pas le temps d'attente dans une file d'attente.
query_cpu _usage_percent	numeric(3 8,2)	Pourcentage de la capacité de l'UC utilisée par la requête.
query_tem p_blocks_to_disk	bigint	Quantité d'espace disque utilisé par une requête pour écrire des résultats intermédiaires, en Mo.
segment_e xecution_time	bigint	Délai écoulé pour l'exécution d'un seul segment (en secondes).
cpu_skew	numeric(3 8,2)	Ratio de l'utilisation maximale de l'UC pour une tranche afin d'obtenir l'utilisation moyenne de l'UC pour toutes les tranches. Cette métrique est définie au niveau du segment.
io_skew	numeric(3 8,2)	Ratio du nombre maximal de blocs lus (I/O) pour une tranche quelconque afin d'obtenir le nombre moyen de blocs lus pour toutes les tranches.

Nom de la colonne	Type de données	Description
scan_row_count	bigint	Nombre de lignes dans une étape d'analyse. Le nombre de lignes correspond au nombre total de lignes émises avant le filtrage des lignes marquées pour la suppression (lignes fantôme) et avant l'application des filtres de requête définis par l'utilisateur.
join_row_count	bigint	Nombre de lignes traitées dans une étape de jonction.
nested_loop_join_row_count	bigint	Nombre de lignes dans une jonction de boucles imbriquées.
return_row_count	bigint	Nombre de lignes retournées par la requête.
spectrum_scan_row_count	bigint	Le nombre de lignes analysées par Amazon Redshift Spectrum dans Amazon S3.
spectrum_scan_size_mb	bigint	Quantité de données, en Mo, analysées par Amazon Redshift Spectrum dans Amazon S3.
query_queue_time	bigint	Durée, en secondes, pendant laquelle la requête a été mise en file d'attente.

SVL_QUERY_QUEUE_INFO

Résume les détails des requêtes ayant passé du temps dans une file d'attente de requête de gestion de la charge de travail (WLM) ou dans une file d'attente de validation.

La vue SVL_QUERY_QUEUE_INFO filtre les requêtes exécutées par le système et affiche uniquement les requêtes exécutées par un utilisateur.

La vue SVL_QUERY_QUEUE_INFO résume les informations des tables système STL_QUERY, STL_WLM_QUERY et STL_COMMIT_STATS.

SVL_QUERY_QUEUE_INFO n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
database	text	Nom de la base de données à laquelle l'utilisateur était connecté lorsque la requête a été émise.
query	entier	ID de requête.
xid	bigint	ID de transaction.
userid	entier	ID de l'utilisateur qui a généré la requête.
querytxt	text	100 premiers caractères du texte de la requête.
queue_start_time	timestamp	Heure UTC à laquelle la requête est entrée dans la file d'attente WLM.
exec_start_time	timestamp	Heure UTC à laquelle l'exécution de la requête a démarré.
service_class	entier	ID de la classe de service. Les classes de service sont définies dans le fichier de configuration WLM.
slots	entier	Nombre d'emplacements de requête WLM.
queue_elapsed	bigint	Temps passée par la requête dans une file d'attente WLM (en secondes).
exec_elapsed	bigint	Temps passé à l'exécution de la requête (en secondes).
wlm_total_elapsed	bigint	Temps que la requête a passé dans une file d'attente WLM (queue_elapsed), plus le temps passé à l'exécution de la requête (exec_elapsed).

Nom de la colonne	Type de données	Description
commit_queue_elapsed	bigint	Temps que la requête a passé à attendre dans la file d'attente de validation (en secondes).
commit_exec_time	bigint	Temps que la requête a passé dans l'opération de validation (en secondes).
service_class_name	character(64)	Nom de la classe de service.

Exemples de requêtes

L'exemple suivant montre le temps que les requêtes ont passé dans les files d'attente WLM.

```
select query, service_class, queue_elapsed, exec_elapsed, wlm_total_elapsed
from svl_query_queue_info
where wlm_total_elapsed > 0;

  query  | service_class | queue_elapsed | exec_elapsed | wlm_total_elapsed
---------+---------------+---------------+--------------+-------------------
 2742669 |             6 |             2 |          916 |               918
 2742668 |             6 |             4 |          197 |               201
(2 rows)
```

SVL_QUERY_REPORT

Amazon Redshift crée la vue SVL_QUERY_REPORT à partir de l'UNION d'un certain nombre de tables système STL Amazon Redshift pour fournir des informations sur les étapes de la requête exécutées.

Cette vue décompose les informations sur les requêtes exécutées par tranche et par étape, ce qui peut aider à dépanner les problèmes de nœud et tranche du cluster Amazon Redshift.

SVL_QUERY_REPORT est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. Permet de joindre d'autres tables système et vues.
slice	entier	Tranche de données où l'étape a été exécutée.
segment	entier	Numéro de segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Les segments de requête peuvent s'exécuter en parallèle. Chaque segment s'exécute dans un processus unique.
étape	entier	Étape de requête terminée.
start_time	timestamp	Heure exacte (au format UTC) de début d'exécution du segment, avec 6 chiffres de précision pour les fractions de seconde. Par exemple : **`2012-12-12 11:29:19.131358`** .
end_time	timestamp	Heure exacte (au format UTC) de fin d'exécution du segment, avec 6 chiffres de précision pour les fractions de seconde. Par exemple : **`2012-12-12 11:29:19.131467`**
elapsed_time	bigint	Durée (en microsecondes) de l'exécution du segment.
rows	bigint	Nombre de lignes générées par l'étape (par tranche). Ce nombre représente le nombre de lignes pour la tranche qui résultent de l'exécution de l'étape, pas le nombre de lignes reçues ou traitées par l'étape. En

Nom de la colonne	Type de données	Description
		d'autres termes, il s'agit du nombre de lignes qui survivent à l'étape et sont passées à l'étape suivante.
bytes	bigint	Nombre d'octets générés par l'étape (par tranche).
étiquette	char(256)	Étiquette de l'étape, qui se compose d'un nom d'étape de requête et, le cas échéant, d'un ID de table et d'un nom de table (par exemple, `scan tbl=100448 name =user`). Les tables à trois chiffres font IDs généralement référence à des scans de tables transitoires. Lorsque `tbl=0` s'affiche, cela fait généralement référence à une analyse d'une valeur constante.
is_diskbased	character (1)	Si cette étape de la requête a été exécutée comme une opération sur disque : true (**t**) ou false (**f**). Seules certaines étapes, telles que le hachage, le tri et l'agrégation, peuvent accéder au disque. La plupart des types d'étapes sont toujours exécutés en mémoire.
workmem	bigint	Quantité de mémoire de travail (en octets) attribuée à l'étape de requête. Cette valeur est le seuil query_working_mem alloué pour l'utilisation lors de l'exécution, pas la quantité de mémoire réellement utilisée
is_rrscan	character (1)	Si la valeur est définie sur true (**t**), indique qu'une analyse à plage restreinte a été utilisée sur l'étape.
is_delayed_scan	character (1)	Si la valeur est définie sur true (**t**), indique qu'une analyse retardée a été utilisée sur l'étape.
rows_pre_filter	bigint	Pour les analyses de tables permanentes, le nombre total de lignes émises avant le filtrage des lignes marquées pour la suppression (lignes fantôme) et avant l'application des filtres de requête définis par l'utilisateur.

Exemples de requêtes

La requête suivante illustre le delta des données des lignes retournées pour la requête avec l'ID 279. Utilisez cette requête pour déterminer si les données de base de données sont réparties de façon uniforme sur les tranches du cluster d'entrepôt des données :

```
select query, segment, step, max(rows), min(rows),
case when sum(rows) > 0
then ((cast(max(rows) -min(rows) as float)*count(rows))/sum(rows))
else 0 end
from svl_query_report
where query = 279
group by query, segment, step
order by segment, step;
```

Cette requête doit renvoyer des données similaires à l'exemple de sortie suivant :

```
query | segment | step |   max    |   min    |         case
------+---------+------+----------+----------+----------------------
279 |       0 |    0 | 19721687 | 19721687 |                    0
279 |       0 |    1 | 19721687 | 19721687 |                    0
279 |       1 |    0 |   986085 |   986084 | 1.01411202804304e-06
279 |       1 |    1 |   986085 |   986084 | 1.01411202804304e-06
279 |       1 |    4 |   986085 |   986084 | 1.01411202804304e-06
279 |       2 |    0 |  1775517 |   788460 |     1.00098637606408
279 |       2 |    2 |  1775517 |   788460 |     1.00098637606408
279 |       3 |    0 |  1775517 |   788460 |     1.00098637606408
279 |       3 |    2 |  1775517 |   788460 |     1.00098637606408
279 |       3 |    3 |  1775517 |   788460 |     1.00098637606408
279 |       4 |    0 |  1775517 |   788460 |     1.00098637606408
279 |       4 |    1 |  1775517 |   788460 |     1.00098637606408
279 |       4 |    2 |        1 |        1 |                    0
279 |       5 |    0 |        1 |        1 |                    0
279 |       5 |    1 |        1 |        1 |                    0
279 |       6 |    0 |       20 |       20 |                    0
279 |       6 |    1 |        1 |        1 |                    0
279 |       7 |    0 |        1 |        1 |                    0
279 |       7 |    1 |        0 |        0 |                    0
(19 rows)
```

SVL_QUERY_SUMMARY

Utilisez la vue SVL_QUERY_SUMMARY pour rechercher des informations générales sur l'exécution d'une requête.

La vue SVL_QUERY_SUMMARY contient un sous-ensemble de données de la vue SVL_QUERY_REPORT. Notez que les informations de SVL_QUERY_SUMMARY sont regroupées à partir de tous les nœuds.

> **Note**
>
> La vue SVL_QUERY_SUMMARY contient uniquement des informations sur les requêtes exécutées par Amazon Redshift, et non sur d'autres commandes DDL ou d'utilitaire. Pour obtenir une liste complète et des informations sur toutes les instructions exécutées par Amazon Redshift, y compris les commandes DDL et d'utilitaire, vous pouvez interroger la vue SVL_STATEMENTTEXT.

SVL_QUERY_SUMMARY est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Pour plus d'informations sur SVCS_QUERY_SOMMY, consultez SVCS_QUERY_SUMMARY.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
query	entier	ID de requête. Permet de joindre d'autres tables système et vues.

Nom de la colonne	Type de données	Description
stm	entier	Flux : ensemble de segments simultanés d'une requête. Une requête possède un ou plusieurs flux.
seg	entier	Numéro de segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Les segments de requête peuvent s'exécuter en parallèle. Chaque segment s'exécute dans un processus unique.
étape	entier	Étape de la requête exécutée.
maxtime	bigint	Durée maximale (en microsecondes) d'exécution de l'étape.
avgtime	bigint	Durée moyenne (en microsecondes) d'exécution de l'étape.
rows	bigint	Nombre de lignes de données impliquées dans l'étape de requête.
octets	bigint	Nombre d'octets de données impliqués dans l'étape de requête.
rate_row	double precision	Taux d'exécution de la requête par ligne.
rate_byte	double precision	Taux d'exécution de la requête par octet.
étiquette	text	Étiquette de l'étape, qui se compose d'un nom d'étape de requête et, le cas échéant, d'un ID de table et d'un nom de table (par exemple, scan tbl=100448 name =user). Les tables à trois chiffres font IDs généralement référence à des scans de tables transitoires. Lorsque `tbl=0` s'affiche, cela fait généralement référence à une analyse d'une valeur constante.
is_diskbased	character(1)	Indique si cette étape de la requête a été exécutée comme opération sur disque sur un nœud du cluster : true (**t**) ou false (**f**). Seules certaines étapes, telles que le hachage, le tri et l'agrégation, peuvent accéder au disque. La plupart des types d'étapes sont toujours exécutés en mémoire.

Nom de la colonne	Type de données	Description
workmem	bigint	Quantité de mémoire de travail (en octets) attribuée à l'étape de requête.
is_rrscan	character(1)	Si la valeur est définie sur true (**t**), indique qu'une analyse à plage restreinte a été utilisée sur l'étape. La valeur par défaut est false (**f**).
is_delayed_scan	character(1)	Si la valeur est définie sur true (**t**), indique qu'une analyse retardée a été utilisée sur l'étape. La valeur par défaut est false (**f**).
rows_pre_filter	bigint	Pour les analyses des tables permanentes, le nombre total de lignes émises avant le filtrage des lignes marquées comme devant être supprimées (lignes fantôme).

Exemples de requêtes

Affichage des informations de traitement d'une étape de requête

La requête suivante affiche les informations de traitement de base de chaque étape de la requête 87 :

```
select query, stm, seg, step, rows, bytes
from svl_query_summary
where query = 87
order by query, seg, step;
```

Cette requête extrait les informations de traitement de la requête 87, comme illustré dans l'exemple de sortie suivant :

```
 query | stm | seg | step |  rows  |  bytes
-------+-----+-----+------+--------+---------
87     |   0 |   0 |    0 |     90 |    1890
87     |   0 |   0 |    2 |     90 |     360
87     |   0 |   1 |    0 |     90 |     360
87     |   0 |   1 |    2 |     90 |    1440
87     |   1 |   2 |    0 | 210494 | 4209880
87     |   1 |   2 |    3 |  89500 |       0
87     |   1 |   2 |    6 |      4 |      96
```

```
87     |   2 |   3 |    0 |      4 |      96
87     |   2 |   3 |    1 |      4 |      96
87     |   2 |   4 |    0 |      4 |      96
87     |   2 |   4 |    1 |      1 |      24
87     |   3 |   5 |    0 |      1 |      24
87     |   3 |   5 |    4 |      0 |       0
(13 rows)
```

Détermination si les étapes de requête ont été répandues sur le disque

La requête suivante indique si l'une des étapes de la requête avec l'ID de requête 1025 (voir la vue SVL_QLOG pour apprendre à obtenir l'ID d'une requête) a été répandue sur le disque ou si la requête a été entièrement exécutée en mémoire :

```
select query, step, rows, workmem, label, is_diskbased
from svl_query_summary
where query = 1025
order by workmem desc;
```

Cette requête renvoie l'exemple de sortie suivant :

```
query| step|  rows  |  workmem   |  label        | is_diskbased
-----+-----+--------+------------+---------------+--------------
1025 |  0  |16000000|  141557760 |scan tbl=9     | f
1025 |  2  |16000000|  135266304 |hash tbl=142   | t
1025 |  0  |16000000|  128974848 |scan tbl=116536| f
1025 |  2  |16000000|  122683392 |dist           | f
(4 rows)
```

En analysant les valeurs d'IS_DISKBASED, vous pouvez voir quelles étapes de requête sont allées sur le disque. Pour la requête 1025, l'étape de hachage a été exécutée sur le disque. Les étapes susceptibles de s'exécuter sur disque incluent les étapes de hachage, d'agrégation et de tri. Pour afficher uniquement les étapes de requête sur disque, ajoutez la clause **`and is_diskbased = 't'`** à l'instruction SQL de l'exemple ci-dessus.

SVL_RESTORE_ALTER_TABLE_PROGRESS

Utilisez SVL_RESTORE_ALTER_TABLE_PROGRESS pour surveiller la progression de la migration de chaque table du cluster lors d'un redimensionnement classique vers des nœuds. RA3 Elle capture le débit historique de la migration des données lors de l'opération de redimensionnement.

Pour plus d'informations sur le redimensionnement classique en RA3 nœuds, consultez la section Redimensionnement classique.

SVL_RESTORE_ALTER_TABLE_PROGRESS n'est visible que par les superutilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_RESTORE_LOG. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

> **Note**
>
> Les lignes dont la progression est égale `100.00%` ou `ABORTED` sont supprimées au bout de 7 jours. Les lignes des tables supprimées pendant ou après un redimensionnement classique peuvent toujours apparaître dans SVL_RESTORE_ALTER_TABLE_PROGRESS.

Colonnes de la table

Nom de la colonne	Type de données	Description
tbl	entier	ID de la table.
progress	char(32)	Statut de progression de la redistribution de la table. Les valeurs possibles sont les pourcentages compris entre `0.00%` `100.00%` et et le message`ABORTED`. `ABORTED`signifie que la redistribution a été arrêtée sans être terminée, pour la raison expliquée dans la `message` colonne.
message	char(256)	Message associé à la progression de la redistribution de la table.

Exemple de requête

La requête suivante renvoie les requêtes en cours d'exécution et en file d'attente.

```
select * from svl_restore_alter_table_progress;
```

```
tbl     | progress |                          message
--------+----------+-----------------------------------------------------------
105614  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105610  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105594  | 0.00%    | Table waiting for alter diststyle conversion.
105602  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105606  | ABORTED  | Abort:Table no longer contains the prior dist key column.
105598  | 100.00%  | Restored to distkey successfully.
```

SVL_S3LIST

Utilisez la vue SVL_S3LIST afin d'obtenir des détails sur les requêtes Amazon Redshift Spectrum au niveau du segment.

SVL_S3LIST est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> SVL_S3LIST contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_EXTERNAL_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
query	entier	ID de requête.
segment	entier	Numéro du segment. Requête composée de plusieurs segments.
node	entier	Numéro du nœud.

Nom de la colonne	Type de données	Description
slice	entier	Tranche de données par rapport à laquelle un segment particulier est exécuté.
eventtime	timestamp	L'heure UTC à laquelle l'événement est enregistré.
bucket	text	Nom du compartiment Amazon S3.
prefix	text	Préfixe de l'emplacement du compartiment Amazon S3.
recursive	char(1)	Si l'analyse des sous-dossiers est récursive.
retrieved_files	entier	Nombre de fichiers répertoriés.
max_file_size	bigint	Taille de fichier maximum parmi les fichiers répertoriés.
avg_file_size	double precision	Taille de fichier moyenne parmi les fichiers répertoriés.
generated_splits	entier	Nombre de fichiers séparés.
avg_split_length	double precision	Longueur moyenne des fichiers séparés en octets.
duration	bigint	Durée de l'établissement de la liste de fichiers en microsecondes.

Exemple de requête

L'exemple suivant interroge SVL_S3LIST par rapport à la dernière requête exécutée.

```
select *
from svl_s3list
where query = pg_last_query_id()
order by query,segment;
```

SVL_S3LOG

Utilisez la vue SVL_S3LOG afin d'obtenir des détails sur les requêtes Amazon Redshift Spectrum au niveau du segment et de la tranche de nœud.

SVL_S3LOG est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> SVL_S3LOG contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_EXTERNAL_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
pid	entier	ID du processus.
query	entier	ID de requête.
segment	entier	Numéro du segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes.
étape	entier	Étape de la requête exécutée.
node	entier	Numéro du nœud.
slice	entier	Tranche de données par rapport à laquelle un segment particulier est exécuté.

Nom de la colonne	Type de données	Description
eventtime	timestamp	Heure UTC à laquelle l'étape a commencé l'exécution.
message	text	Message pour l'entrée de journal.

Exemple de requête

L'exemple suivant interroge SVCS_S3LOG par rapport à la dernière requête exécutée.

```
select *
from svl_s3log
where query = pg_last_query_id()
order by query,segment,slice;
```

SVL_S3PARTITION

Utilisez la vue SVL_S3PARTITION afin d'obtenir des détails sur les partitions Amazon Redshift Spectrum au niveau du segment et de la tranche de nœud.

SVL_S3PARTITION est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> SVL_S3PARTITION contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_EXTERNAL_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
query	entier	ID de requête.
segment	entier	Numéro de segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes.
node	entier	Numéro du nœud.
slice	entier	Tranche de données par rapport à laquelle un segment particulier est exécuté.
starttime	horodatage sans fuseau horaire	Heure UTC à laquelle le nettoyage de la partition a commencé à s'exécuter.
endtime	horodatage sans fuseau horaire	Heure UTC à laquelle le nettoyage de la partition s'est terminé.
duration	bigint	Temps écoulé (en microsecondes).
total_partitions	entier	Nombre total de partitions.
qualified_partitions	entier	Nombre de partitions qualifiées.
assigned_partitions	entier	Nombre de partitions affectées sur la tranche.
assignment	character	Type d'affectation.

Exemple de requête

L'exemple suivant permet d'obtenir les détails de la partition pour la dernière requête exécutée.

```
SELECT query, segment,
       MIN(starttime) AS starttime,
       MAX(endtime) AS endtime,
       datediff(ms,MIN(starttime),MAX(endtime)) AS dur_ms,
       MAX(total_partitions) AS total_partitions,
       MAX(qualified_partitions) AS qualified_partitions,
       MAX(assignment) as assignment_type
FROM svl_s3partition
WHERE query=pg_last_query_id()
GROUP BY query, segment
```

```
query | segment |           starttime           |           endtime           | dur_ms|
 total_partitions | qualified_partitions | assignment_type
------+---------+-------------------------------+-----------------------------+-------
+------------------+----------------------+----------------
99232 |       0 | 2018-04-17 22:43:50.201515    | 2018-04-17 22:43:54.674595  |  4473 |
       2526       |        334           | p
```

SVL_S3PARTITION_SUMMARY

Utilisez la vue SVL_S3PARTITION_SUMMARY pour obtenir un résumé du traitement de la partition des requêtes Redshift Spectrum au niveau du segment.

SVL_S3PARTITION_SUMMARY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Pour plus d’informations sur SVCS_S3PARTITION, consultez SVCS_S3PARTITION_SUMMARY.

Colonnes de la table

Nom de la colonne	Type de données	Description
query	entier	ID de requête. Vous pouvez utiliser cette valeur pour joindre d’autres vues et tables système.
segment	entier	Numéro du segment. Requête composée de plusieurs segments.

Nom de la colonne	Type de données	Description
assignment	char(1)	Type d'attribution de la partition entre les nœuds.
min_starttime	timestamp	Heure UTC à laquelle le traitement de la partition a commencé.
max_endtime	timestamp	Heure UTC à laquelle le traitement de la partition s'est terminé.
min_duration	bigint	Durée minimum de traitement de la partition utilisée par un nœud pour cette requête (en microsecondes).
max_duration	bigint	Durée maximum de traitement de la partition utilisée par un nœud pour cette requête (en microsecondes).
avg_duration	bigint	Durée moyenne de traitement de la partition utilisée par un nœud pour cette requête (en microsecondes).
total_partitions	entier	Nombre total de partitions dans une table externe.
qualified_partitions	entier	Nombre total de partitions qualifiées.
min_assigned_partitions	entier	Nombre minimum de partitions attribuées sur un nœud.
max_assigned_partitions	entier	Nombre maximum de partitions attribuées sur un nœud.
avg_assigned_partitions	bigint	Nombre moyen de partitions attribuées sur un nœud.

Exemple de requête

L'exemple suivant permet d'obtenir les détails de l'analyse de la partition pour la dernière requête exécutée.

```
select query, segment, assignment, min_starttime, max_endtime, min_duration,
 avg_duration
from svl_s3partition_summary
where query = pg_last_query_id()
order by query,segment;
```

SVL_S3QUERY

Utilisez la vue SVL_S3QUERY afin d'obtenir des détails sur les requêtes Amazon Redshift Spectrum au niveau du segment et de la tranche de nœud.

SVL_S3QUERY est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

> **Note**
>
> SVL_S3QUERY contient uniquement les requêtes exécutées sur les principaux clusters provisionnés. Il ne contient pas de requêtes exécutées sur des clusters de dimensionnement simultané ou sur des espaces de noms sans serveur. Pour accéder aux plans d'explication des requêtes exécutées à la fois sur les clusters principaux, les clusters de dimensionnement de la simultanéité et les espaces de noms sans serveur, nous vous recommandons d'utiliser la vue de surveillance SYS. SYS_EXTERNAL_QUERY_DETAIL Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré une entrée donnée.
query	entier	ID de requête.

Nom de la colonne	Type de données	Description
segment	entier	Numéro de segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes.
étape	entier	Étape de la requête exécutée.
node	entier	Numéro du nœud.
slice	entier	Tranche de données par rapport à laquelle un segment particulier est exécuté.
starttime	timestamp	Heure UTC de début d'exécution de la requête.
endtime	timestamp	Heure UTC de fin d'exécution de la requête.
elapsed	entier	Temps écoulé (en microsecondes).
external_table_name	char(136)	Format interne du nom de la table externe pour l'étape d'analyse Amazon S3.
is_partitioned	char(1)	Si cette valeur de colonne est true (**t**), indique que la table externe est partitionnée.
is_rrscan	char(1)	Si cette valeur de colonne est true (**t**), indique qu'une analyse à plage restreinte a été appliquée.
s3_scanned_rows	bigint	Nombre de lignes analysées à partir d'Amazon S3 et envoyées à la couche Redshift Spectrum.
s3_scanned_bytes	bigint	Nombre d'octets analysés à partir d'Amazon S3 et envoyés à la couche Redshift Spectrum.
s3query_returned_rows	bigint	Nombre de lignes retournées par la couche Redshift Spectrum au cluster.

Nom de la colonne	Type de données	Description
s3query_returned_bytes	bigint	Nombre d'octets retournés par la couche Redshift Spectrum au cluster.
fichiers	entier	Nombre de fichiers traités pour cette étape d'analyse S3 sur cette tranche.
splits	int	Nombre de divisions traitées sur cette tranche. Avec des fichiers de données divisibles volumineux, par exemple, des fichiers de données supérieurs à environ 512 Mo, Redshift Spectrum essaie de diviser les fichiers en plusieurs demandes S3 de traitement parallèle.
total_split_size	bigint	Taille totale de toutes les divisions traitées sur cette tranche, en octets.
max_split_size	bigint	Taille de division maximale traitée pour cette tranche, en octets.
total_retries	entier	Nombre total de nouvelles tentatives pour les fichiers traités.
max_retries	entier	Nombre maximal de nouvelles tentatives pour un fichier traité spécifique.
max_request_duration	entier	Durée maximum d'une demande Redshift Spectrum spécifique (en microsecondes).
avg_request_duration	double precision	Durée moyenne des demandes Redshift Spectrum (en microsecondes).
max_request_parallelism	entier	Nombre maximal de requêtes Redshift Spectrum en attente sur cette tranche pour cette étape d'analyse S3.

Nom de la colonne	Type de données	Description
avg_request_parallelism	double precision	Nombre moyen de demandes Redshift Spectrum parallèles sur cette tranche pour cette étape d'analyse S3.

Exemple de requête

L'exemple suivant permet d'obtenir les détails de l'étape d'analyse pour la dernière requête exécutée.

```
select query, segment, slice, elapsed, s3_scanned_rows, s3_scanned_bytes,
 s3query_returned_rows, s3query_returned_bytes, files
from svl_s3query
where query = pg_last_query_id()
order by query,segment,slice;
```

```
query | segment | slice | elapsed | s3_scanned_rows | s3_scanned_bytes |
 s3query_returned_rows | s3query_returned_bytes | files
------+---------+-------+---------+-----------------+------------------
+----------------------+-------------------------+------
 4587 |       2 |     0 |   67811 |               0 |                0 |
      0 |                      0 |     0
 4587 |       2 |     1 |  591568 |          172462 |         11260097 |
   8513 |                  170260 |     1
 4587 |       2 |     2 |  216849 |               0 |                0 |
      0 |                      0 |     0
 4587 |       2 |     3 |  216671 |               0 |                0 |
      0 |                      0 |     0
```

SVL_S3QUERY_SUMMARY

Utilisez la vue SVL_S3QUERY_SUMMARY pour obtenir un résumé de toutes les requêtes Amazon Redshift Spectrum (requêtes Amazon S3) qui ont été exécutées sur le système. SVL_S3QUERY_SUMMARY regroupe les détails issus de SVL_S3QUERY au niveau du segment.

SVL_S3QUERY_SUMMARY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_EXTERNAL_QUERY_DETAIL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Pour SVCS_S3QUERY_SOMMY, consultez SVCS_S3QUERY_SUMMARY.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée donnée.
query	entier	ID de requête. Vous pouvez utiliser cette valeur pour joindre d'autres vues et tables système.
xid	bigint	ID de transaction.
pid	entier	ID du processus.
segment	entier	Numéro du segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes.
étape	entier	Étape de la requête exécutée.
starttime	timestamp	Heure UTC de début d'exécution de la requête.
endtime	timestamp	Heure UTC de fin d'exécution de la requête.
elapsed	entier	Durée nécessaire à l'exécution de la requête (en microsecondes).
aborted	entier	Si une requête a été arrêtée par le système ou annulée par l'utilisateur, cette colonne contient **1**. Si la requête est terminée, cette colonne contient **0**.

Nom de la colonne	Type de données	Description
external_table_name	char(136)	Format interne du nom externe de la table pour l'analyse de la table externe.
file_format	character(16)	Format de fichier des données de la table externe.
is_partitioned	char(1)	Si cette valeur de colonne est true (**t**), indique que la table externe est partitionnée.
is_rrscan	char(1)	Si cette valeur de colonne est true (**t**), indique qu'une analyse à plage restreinte a été appliquée.
is_nested	char(1)	Si cette valeur de colonne est true (**t**), indique que le type de données de la colonne imbriquée est accessible.
s3_scanned_rows	bigint	Nombre de lignes analysées à partir d'Amazon S3 et envoyées à la couche Redshift Spectrum.
s3_scanned_bytes	bigint	Nombre d'octets analysés à partir d'Amazon S3 et envoyés à la couche Redshift Spectrum, en fonction des données compressées.
s3query_returned_rows	bigint	Nombre de lignes retournées par la couche Redshift Spectrum au cluster.
s3query_returned_bytes	bigint	Nombre d'octets retournés par la couche Redshift Spectrum au cluster. Si le volume des données renvoyées à Amazon Redshift est important, les performances du système peuvent être affectées.
fichiers	entier	Nombre de fichiers traités pour cette requête Redshift Spectrum. Les avantages du traitement parallèle sont amoindris s'il y a peu de fichiers.
files_max	entier	Nombre maximal de fichiers traités sur une tranche.

Nom de la colonne	Type de données	Description
files_avg	entier	Nombre moyen de fichiers traités sur une tranche.
splits	int	Nombre de divisions traitées pour ce segment. Nombre de divisions traitées sur cette tranche. Avec des fichiers de données divisibles volumineux, par exemple, des fichiers de données supérieurs à environ 512 Mo, Redshift Spectrum essaie de diviser les fichiers en plusieurs demandes S3 de traitement parallèle.
splits_max	int	Nombre maximal de divisions traitées sur cette tranche.
splits_avg	int	Nombre moyen de divisions traitées sur cette tranche.
total_split_size	bigint	Taille totale de toutes les divisions traitées.
max_split_size	bigint	Taille de division maximale traitée, en octets.
avg_split_size	bigint	Taille de division moyenne traitée, en octets.
total_retries	entier	Nombre total de nouvelles tentatives pour un fichier traité spécifique.
max_retries	entier	Nombre maximal de nouvelles tentatives pour n’importe lequel des fichiers traités.
max_request_duration	entier	Durée maximale d'une requête de fichier spécifique (en microsecondes). Les requêtes de longue durée peuvent indiquer la présence d’un goulot d’étranglement.
avg_request_duration	double precision	Durée moyenne des requêtes de fichier (en microsecondes).

Nom de la colonne	Type de données	Description
max_request_parallelism	entier	Nombre maximum de demandes parallèles sur une tranche pour cette requête Redshift Spectrum.
avg_request_parallelism	double precision	Nombre moyen de demandes parallèles sur une tranche pour cette requête Redshift Spectrum.
total_slowdown_count	bigint	Nombre total de demandes Amazon S3 avec une erreur de ralentissement qui se produit lors de l'analyse de la table externe.
max_slowdown_count	entier	Nombre maximum de demandes Amazon S3 avec une erreur de ralentissement qui se produit lors de l'analyse de la table externe sur une tranche.

Exemple de requête

L'exemple suivant permet d'obtenir les détails de l'étape d'analyse pour la dernière requête exécutée.

```
select query, segment, elapsed, s3_scanned_rows, s3_scanned_bytes,
 s3query_returned_rows, s3query_returned_bytes, files
from svl_s3query_summary
where query = pg_last_query_id()
order by query,segment;
```

```
query | segment | elapsed | s3_scanned_rows | s3_scanned_bytes | s3query_returned_rows
 | s3query_returned_bytes | files
------+---------+---------+-----------------+------------------+----------------------
+------------------------+------
 4587 |       2 |   67811 |               0 |                0 |                    0
 |                      0 |     0
 4587 |       2 |  591568 |          172462 |         11260097 |                 8513
 |                 170260 |     1
 4587 |       2 |  216849 |               0 |                0 |                    0
 |                      0 |     0
```

```
 4587 |       2 |  216671 |               0 |               0 |                     0
 |                     0 |     0
```

SVL_S3RETRIES

Utilisez la vue SVL_S3RETRIES pour savoir pourquoi une requête Amazon Redshift Spectrum basée sur Amazon S3 a échoué.

SVL_S3RETRIES est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
query	entier	ID de requête.
segment	entier	Numéro de segment. Une requête se compose de plusieurs segments et chaque segment d'une ou de plusieurs étapes. Les segments de requête peuvent s'exécuter en parallèle. Chaque segment s'exécute dans un processus unique.
node	entier	Numéro du nœud.
slice	entier	Tranche de données par rapport à laquelle un segment particulier est exécuté.

Nom de la colonne	Type de données	Description
eventtime	horodatage sans fuseau horaire	Heure UTC à laquelle l'étape a commencé l'exécution.
retries	entier	Nombre de nouvelles tentatives pour la requête.
successful_fetches	entier	Nombre de fois où les données ont été renvoyées.
file_size	bigint	Taille du fichier en octets.
location	text	Emplacement de la table.
message	text	Message d'erreur.

Exemple de requête

L'exemple suivant extrait les données relatives aux requêtes S3 qui ont échoué.

```
SELECT svl_s3retries.query, svl_s3retries.segment, svl_s3retries.node,
 svl_s3retries.slice, svl_s3retries.eventtime, svl_s3retries.retries,
svl_s3retries.successful_fetches, svl_s3retries.file_size,
 btrim((svl_s3retries."location")::text) AS "location",
 btrim((svl_s3retries.message)::text)
AS message FROM svl_s3retries;
```

SVL_SPATIAL_SIMPLIFICATION

Vous pouvez interroger la vue système SVL_SPATIAL_SIMPLIFY pour obtenir des informations sur les objets de géométrie spatiale simplifiée à l'aide de la commande COPY. Lorsque vous utilisez

COPY sur un fichier de formes, vous pouvez spécifier les options d'ingestion SIMPLIFY `tolerance`, SIMPLIFY AUTO et SIMPLIFY AUTO `max_tolerance`. Le résultat de la simplification est résumé dans la vue système SVL_SPATIAL_SIMPLIFY.

Lorsque SIMPLIFY AUTO `max_tolerance` est défini, cette vue contient une ligne pour chaque géométrie ayant dépassé la taille maximale. Lorsque SIMPLIFY `tolerance` est défini, une ligne pour l'ensemble de l'opération COPY est stockée. Cette ligne fait référence à l'ID de la requête COPY et à la tolérance de simplification spécifiée.

SVL_SPATIAL_SIMPLIFY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_SPATIAL_SIMPLIFY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
query	entier	L'ID de la requête (commande COPY) qui a généré cette ligne.
line_number	entier	Lorsque l'option `SIMPLIFY AUTO` COPY est spécifiée , cette valeur est le numéro de l'enregistrement simplifié dans le fichier de formes.
maximum_tolerance	double	La valeur de tolérance de distance spécifiée dans la commande COPY. Il s'agit soit de la valeur de tolérance maximale via l'option `SIMPLIFY AUTO`, soit de la valeur de tolérance fixe via l'option `SIMPLIFY`.
initial_size	entier	La taille en octets de la valeur de données `GEOMETRY` avant simplification.

Nom de la colonne	Type de données	Description
simplified	char(1)	Lorsque l'option SIMPLIFY AUTO COPY est spécifiée, la valeur indique t si la géométrie a été simplifiée ou f si ce n'est pas le cas. La géométrie peut ne pas être simplifié e si, après la simplification avec la tolérance maximale donnée, la taille est toujours supérieure à la taille maximale de la géométrie.
final_size	entier	Lorsque l'option SIMPLIFY AUTO COPY est spécifiée, il s'agit de la taille en octets de la géométrie après simplific ation.
final_tol erance	double	

Exemple de requête

La requête suivante renvoie la liste des enregistrements simplifiés par COPY.

```
SELECT * FROM svl_spatial_simplify WHERE query = pg_last_copy_id();
 query | line_number | maximum_tolerance | initial_size | simplified | final_size |
 final_tolerance
-------+-------------+-------------------+--------------+------------+------------
+----------------------
    20 |     1184704 |                -1 |      1513736 | t          |    1008808 |
 1.276386653895e-05
    20 |     1664115 |                -1 |      1233456 | t          |    1023584 |
 6.11707814796635e-06
```

SVL_SPECTRUM_SCAN_ERROR

Vous pouvez interroger la vue système SVL_SPECTRUM_SCAN_ERROR pour obtenir des informations sur les erreurs d'analyse Redshift Spectrum.

SVL_SPECTRUM_SCAN_ERROR est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres

données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_EXTERNAL_QUERY_ERROR. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Affiche un exemple d'erreurs enregistrées. La valeur par défaut est de 10 entrées par requête.

Nom de la colonne	Type de données	Description
userid	entier	L'ID de l'utilisateur qui a généré cette ligne.
query	entier	L'ID de la requête qui a généré cette ligne.
location	character(128)	L'emplacement des données à interroger.
rowid	character(128)	Emplacement de l'erreur dans le fichier. Les parties `rowid` sont séparées par `:` (deux-points), et des parties supplémentaires peuvent être ajoutées plus tard. `row_offset :row_group :row_id` Un row_offset correspond au décalage (en octets) de la ligne dans le fichier et est défini sur `-1` pour les formats de fichier non pris en charge. Une table est divisée en row_groups, et chaque groupe possède des lignes avec des row_ids distincts.
colname	character(128)	Le nom de la colonne renvoyée par la requête.
original_value	character(128)	Valeur d'origine interrogée.

Nom de la colonne	Type de données	Description
modified_value	character(128)	Valeur modifiée renvoyée en fonction de l'option de configuration de gestion des données spécifiée dans la requête.
déclencheur	character(128)	Option de gestion des données spécifiée dans la requête.
action	character(128)	Action associée à l'option de gestion des données spécifiée dans la requête.
action_value	character(128)	Paramètre de valeur d'action associé à l'option de traitement des données spécifiée dans la requête.
error_code	entier	Code de résultat de l'option de traitement des données spécifiée dans la requête.

Exemple de requête

La requête suivante renvoie la liste des lignes pour lesquelles des opérations de traitement des données ont été effectuées.

```
SELECT * FROM svl_spectrum_scan_error;
```

La requête renvoie des résultats semblables à ce qui suit.

```
   userid  query     location                                    rowid   colname
          original_value            modified_value       trigger          action
                  action_valueerror_code
     100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0     league_name
          Barclays Premier League    Barclays Premier Lea UNSPECIFIED      TRUNCATE
                                      156
     100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:0     league_nspi
           34595                      32767                UNSPECIFIED
 OVERFLOW_VALUE                                  199
```

```
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:1     league_nspi
          34151                      32767               UNSPECIFIED
OVERFLOW_VALUE                                199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2     league_name
         Barclays Premier League    Barclays Premier Lea UNSPECIFIED      TRUNCATE
                                    156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:2     league_nspi
          33223                      32767               UNSPECIFIED
OVERFLOW_VALUE                                199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3     league_name
         Barclays Premier League    Barclays Premier Lea UNSPECIFIED      TRUNCATE
                                    156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:3     league_nspi
          32808                      32767               UNSPECIFIED
OVERFLOW_VALUE                                199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:4     league_nspi
          32790                      32767               UNSPECIFIED
OVERFLOW_VALUE                                199
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:5     league_name
         Spanish Primera Division   Spanish Primera Divi UNSPECIFIED      TRUNCATE
                                    156
    100   1574007   s3://spectrum-uddh/league/spi_global_rankings.0:6     league_name
         Spanish Primera Division   Spanish Primera Divi UNSPECIFIED      TRUNCATE
                                    156
```

SVL_STATEMENTTEXT

Utilisez la vue SVL_STATEMENTTEXT pour obtenir un enregistrement complet de toutes les commandes SQL qui ont été exécutées sur le système.

La vue SVL_STATEMENTTEXT contient l'union de toutes les lignes des tables STL_DDLTEXT, STL_QUERYTEXT et STL_UTILITYTEXT. Cette vue inclut aussi une jointure avec la table STL_QUERY.

SVL_STATEMENTTEXT est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être

plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur qui a généré l'entrée.
xid	bigint	ID de transaction associé à l'instruction.
pid	entier	ID de processus de l'instruction.
étiquette	caractère (320)	Le nom du fichier utilisé pour exécuter la requête ou une étiquette définie avec une commande SET QUERY_GRO UP. Si la requête n'est pas basée sur un fichier ou si le paramètre QUERY_GROUP n'est pas défini, le champ est vide.
starttime	timestamp	Heure exacte à laquelle l'instruction a démarré l'exécution, avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.131358`** .
endtime	timestamp	Heure exacte à laquelle l'instruction a fini l'exécution, avec 6 chiffres de précision pour les fractions de secondes. Par exemple : **`2009-06-12 11:29:19.193640`** .
sequence	entier	Lorsqu'une seule instruction contient plus de 200 caractère s, des lignes supplémentaires sont enregistrées pour l'instruction. Sequence 0 correspond à la première ligne, 1 à la deuxième, et ainsi de suite.
type	varchar(10)	Type d'instruction SQL : **`QUERY`**, **`DDL`** ou **`UTILITY`**.
text	character(200)	Texte SQL, par incréments de 200 caractères. Ce champ peut contenir des caractères spéciaux tels qu'une barre oblique inverse (\\) et un caractère de saut de ligne (\n).

Exemple de requête

La requête suivante renvoie les instructions DDL qui ont été exécutées le 16 juin 2009 :

```
select starttime, type, rtrim(text) from svl_statementtext
where starttime like '2009-06-16%' and type='DDL' order by starttime asc;

starttime                  | type |              rtrim
---------------------------|------|-------------------------------
2009-06-16 10:36:50.625097 | DDL  | create table ddltest(c1 int);
2009-06-16 15:02:16.006341 | DDL  | drop view alltickitjoin;
2009-06-16 15:02:23.65285  | DDL  | drop table sales;
2009-06-16 15:02:24.548928 | DDL  | drop table listing;
2009-06-16 15:02:25.536655 | DDL  | drop table event;
...
```

Reconstruction de SQL stocké

Pour reconstruire le SQL stocké dans la colonne `text` de SVL_STATEMENTTEXT, exécutez une instruction SELECT pour créer SQL depuis une ou plusieurs parties de la colonne `text`. Avant d'exécuter le SQL reconstruit, remplacez tout caractère spécial (\n) par un saut de ligne. Le résultat de l'instruction SELECT suivante se compose de lignes de SQL reconstruit dans le champ `query_statement`.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS query_statement
from SVL_STATEMENTTEXT where pid=pg_backend_pid();
```

Par exemple, la requête suivante sélectionne 3 colonnes. La requête elle-même contient plus de 200 caractères et est stockée dans plusieurs parties de SVL_STATEMENTTEXT.

```
select
1 AS a01234567890123456789012345678901234567890123456789012345678901234567890,
2 AS b01234567890123456789012345678901234567890123456789012345678901234567890,
3 AS b01234567890123456789012345678901234567890123
FROM stl_querytext;
```

Dans cet exemple, la requête est stockée dans deux parties (lignes) de la colonne `text` de SVL_STATEMENTTEXT.

```
select sequence, text from SVL_STATEMENTTEXT where pid = pg_backend_pid() order by
 starttime, sequence;
```

```
 sequence |
                text

----------
+-----------------------------------------------------------------------------------------------
        0 | select\n1 AS
 a012345678901234567890123456789012345678901234567890123456789012345678901234567890,\n2 AS
 b012345678901234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b012345678901234567890123456789012345678901234
        1 | \nFROM stl_querytext;
```

Pour reconstruire le SQL stocké dans STL_STATEMENTTEXT, exécutez le SQL suivant.

```
select LISTAGG(CASE WHEN LEN(RTRIM(text)) = 0 THEN text ELSE RTRIM(text) END, '')
 within group (order by sequence) AS text
from SVL_STATEMENTTEXT where pid=pg_backend_pid();
```

Pour utiliser le SQL reconstruit qui en résulte dans votre client, remplacez tout caractère spécial (\n) par un saut de ligne.

```
                     text

-----------------------------------------------------------------------------------------------
 select\n1 AS a012345678901234567890123456789012345678901234567890123456789012345678901234567890,
\n2 AS b012345678901234567890123456789012345678901234567890123456789012345678901234567890,\n3 AS
 b012345678901234567890123456789012345678901234\nFROM stl_querytext;
```

SVL_STORED_PROC_CALL

Vous pouvez interroger la vue système SVL_STORED_PROC_CALL pour obtenir des informations sur les appels de procédure stockée, y compris l'heure de début, l'heure de fin et si un appel est abandonné. Chaque appel de procédure stockée reçoit un ID de requête.

SVL_STORED_PROC_CALL est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_PROCEDURE_CALL. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur dont les privilèges ont été utilisés pour exécuter l'instruction. Si cet appel était imbriqué dans une procédure stockée SECURITY DEFINER, il s'agit alors de l'ID utilisateur du propriétaire de cette procédure stockée.
session_userid	entier	ID de l'utilisateur qui a créé la séance et qui est le demandeur de l'appel de procédure stockée de premier niveau.
query	entier	ID de requête de l'appel de procédure.
étiquette	caractère (320)	Le nom du fichier utilisé pour exécuter la requête ou une étiquette définie avec une commande SET QUERY_GROUP. Si la requête n'est pas basée sur un fichier ou si le paramètre QUERY_GROUP n'est pas défini, le champ contient sa valeur par défaut.
xid	bigint	ID de transaction.
pid	entier	ID du processus. En règle générale, toutes les requêtes d'une séance étant exécutées dans le même processus, cette valeur reste constante si vous exécutez une série de requêtes dans la même séance. Suite à certains événements internes, Amazon Redshift peut redémarre

Nom de la colonne	Type de données	Description
		r une séance active et attribuer une nouvelle valeur PID. Pour plus d'informations, consultez STL_RESTARTED_SESSIONS.
database	character(32)	Nom de la base de données à laquelle l'utilisateur était connecté lorsque la requête a été émise.
querytxt	character(4000)	Texte réel de la requête d'appel de procédure.
starttime	timestamp	Heure au format UTC de début d'exécution de la requête, avec une précision de 6 chiffres pour les fractions de secondes. Par exemple : **2009-06-12 11:29:19.131358.**
endtime	timestamp	Heure au format UTC de fin d'exécution de la requête, avec une précision de 6 chiffres pour les fractions de secondes. Par exemple : **2009-06-12 11:29:19.131358.**
aborted	entier	Si une procédure stockée a été arrêtée par le système ou annulée par l'utilisateur, cette colonne contient 1. Si l'appel s'exécute jusqu'à la fin, cette colonne contient 0.
from_sp_call	entier	Si l'appel de procédure a été appelé par un autre appel de procédure, cette colonne contient l'ID de requête de l'appel extérieur. Sinon, le champ a pour valeur NULL.

Exemple de requête

La requête suivante renvoie le temps écoulé dans l'ordre décroissant et le statut d'achèvement des appels de procédure stockée au cours de la dernière journée.

```
select query, datediff(seconds, starttime, endtime) as elapsed_time, aborted,
 trim(querytxt) as call from svl_stored_proc_call where starttime >= getdate() -
 interval '1 day' order by 2 desc;
```

```
  query | elapsed_time | aborted |                                        call
--------+--------------+---------
+-----------------------------------------------------------------------------------
   4166 |            7 |       0 | call search_batch_status(35,'succeeded');
   2433 |            3 |       0 | call test_batch (123456)
   1810 |            1 |       0 | call prod_benchmark (123456)
   1836 |            1 |       0 | call prod_testing (123456)
   1808 |            1 |       0 | call prod_portfolio ('N', 123456)
   1816 |            1 |       1 | call prod_portfolio ('Y', 123456)
```

SVL_STORED_PROC_MESSAGES

Vous pouvez interroger la vue système SVL_STORED_PROC_MESSAGES pour obtenir des informations sur les messages de procédure stockée. Les messages déclenchés sont consignés même si l'appel de procédure stockée est annulé. Chaque appel de procédure stockée reçoit un ID de requête. Pour plus d'informations sur la définition du niveau minimum pour les messages consignés, consultez stored_proc_log_min_messages.

SVL_STORED_PROC_MESSAGES est visible pour tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_PROCEDURE_MESSAGES. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
userid	entier	ID de l'utilisateur dont les privilèges ont été utilisés pour exécuter l'instruction. Si cet appel était imbriqué dans une procédure stockée SECURITY DEFINER, il s'agit alors de l'ID utilisateur du propriétaire de cette procédure stockée.

Nom de la colonne	Type de données	Description
session_userid	entier	ID de l'utilisateur qui a créé la séance et qui est le demandeur de l'appel de procédure stockée de premier niveau.
pid	entier	ID du processus.
xid	bigint	ID de transaction de la requête d'appel de procédure.
query	entier	ID de requête de l'appel de procédure.
recordtime	timestamp	Heure en UTC où le message a été déclenché.
loglevel	entier	Valeur numérique du niveau de journal du message déclenché. Valeurs possibles : 20 — pour JOURNAL 30 — pour INFO 40 — pour AVIS 50 — pour AVERTISSEMENT 60 — pour EXCEPTION
loglevel_text	character(10)	Niveau de journal correspondant à la valeur numérique dans loglevel. Valeurs possibles : LOG, INFO, AVIS, AVERTISSEMENT et EXCEPTION.
message	character(1024)	Le texte du message soulevé.
linenum	entier	Numéro de ligne de l'instruction surélevée.
querytext	character(500)	Texte réel de la requête d'appel de procédure.
étiquette	caractère (320)	Le nom du fichier utilisé pour exécuter la requête ou une étiquette définie avec une commande SET QUERY_GROUP. Si la requête n'est pas basée sur un fichier ou si le paramètre QUERY_GROUP n'est pas défini, le champ contient sa valeur par défaut.
aborted	entier	Si une procédure stockée a été arrêtée par le système ou annulée par l'utilisateur, cette colonne contient 1. Si l'appel s'exécute jusqu'à la fin, cette colonne contient 0.

Nom de la colonne	Type de données	Description
message_xid	bigint	ID de transaction du message généré.

Exemple de requête

Les instructions SQL suivantes montrent comment utiliser SVL_STORED_PROC_MESSAGES pour passer en revue les messages générés.

```
-- Create and run a stored procedure
CREATE OR REPLACE PROCEDURE test_proc1(f1 int) AS
$$
BEGIN
    RAISE INFO 'Log Level: Input f1 is %',f1;
    RAISE NOTICE 'Notice Level: Input f1 is %',f1;
    EXECUTE 'select invalid';
    RAISE NOTICE 'Should not print this';

EXCEPTION WHEN OTHERS THEN
     raise exception 'EXCEPTION level: Exception Handling';
END;
$$ LANGUAGE plpgsql;

-- Call this stored procedure
CALL test_proc1(2);

-- Show raised messages with level higher than INFO
SELECT query, recordtime, loglevel, loglevel_text, trim(message) as message, aborted
 FROM svl_stored_proc_messages
  WHERE loglevel > 30 AND query = 193 ORDER BY recordtime;

 query |         recordtime         | loglevel | loglevel_text |               message
             | aborted
-------+----------------------------+----------+---------------
+-------------------------------------+---------
   193 | 2020-03-17 23:57:18.277196 |       40 | NOTICE        | Notice Level: Input f1
 is 2         |       1
   193 | 2020-03-17 23:57:18.277987 |       60 | EXCEPTION     | EXCEPTION level:
 Exception Handling |       1
```

```
(2 rows)

-- Show raised messages at EXCEPTION level
SELECT query, recordtime, loglevel, loglevel_text, trim(message) as message, aborted
 FROM svl_stored_proc_messages
  WHERE loglevel_text = 'EXCEPTION' AND query = 193 ORDER BY recordtime;

 query |         recordtime         | loglevel | loglevel_text |               message
              | aborted
-------+----------------------------+----------+---------------
+-----------------------------------+---------
   193 | 2020-03-17 23:57:18.277987 |       60 | EXCEPTION     | EXCEPTION level:
 Exception Handling |       1
```

Les instructions SQL suivantes montrent comment utiliser SVL_STORED_PROC_MESSAGES pour passer en revue les messages générés avec l'option SET lors de la création d'une procédure stockée. Étant donné que test_proc() a un niveau de journal minimal AVIS, seuls les messages de niveau AVIS, AVERTISSEMENT et EXCEPTION sont enregistrés dans SVL_STORED_PROC_MESSAGES.

```
-- Create a stored procedure with minimum log level of NOTICE
CREATE OR REPLACE PROCEDURE test_proc() AS
$$
BEGIN
    RAISE LOG 'Raise LOG messages';
    RAISE INFO 'Raise INFO messages';
    RAISE NOTICE 'Raise NOTICE messages';
    RAISE WARNING 'Raise WARNING messages';
    RAISE EXCEPTION 'Raise EXCEPTION messages';
    RAISE WARNING 'Raise WARNING messages again'; -- not reachable
END;
$$ LANGUAGE plpgsql SET stored_proc_log_min_messages = NOTICE;

-- Call this stored procedure
CALL test_proc();

-- Show the raised messages
SELECT query, recordtime, loglevel_text, trim(message) as message, aborted FROM
 svl_stored_proc_messages
  WHERE query = 149 ORDER BY recordtime;

 query |         recordtime         | loglevel_text |          message          |
 aborted
```

```
-------+---------------------------+--------------+-------------------------
+---------
   149 | 2020-03-16 21:51:54.847627 | NOTICE        | Raise NOTICE messages    |
 1
   149 | 2020-03-16 21:51:54.84766  | WARNING       | Raise WARNING messages   |
 1
   149 | 2020-03-16 21:51:54.847668 | EXCEPTION     | Raise EXCEPTION messages |
 1
(3 rows)
```

SVL_TERMINATE

Enregistre l'heure à laquelle un utilisateur annule ou met fin à un processus.

SELECT PG_TERMINATE_BACKEND(pid), SELECT PG_CANCEL_BACKEND(pid) et CANCEL pid crée une entrée de journal dans SVL_TERMINATE.

SVL_ TERMINATE n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_QUERY_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
pid	entier	ID de processus du processus annulé ou résilié.
eventtime	timestamp	Heure à laquelle le processus est annulé ou résilié.
userid	entier	ID utilisateur de l'utilisateur exécutant la commande.
type	chaîne	Type de résiliation. Il s'agit d'ANNULÉ ou de RÉSILIÉ.

La commande suivante affiche la dernière requête annulée.

```
select * from svl_terminate order by eventtime desc limit 1;
```

```
 pid  |         eventtime          | userid |  type
------+----------------------------+--------+--------
 8324 | 2020-03-24 09:42:07.298937 |      1 | CANCEL
(1 row)
```

SVL_UDF_LOG

Enregistre la génération des messages d'erreur et d'avertissement définis par le système au cours de l'exécution de la fonction définie par l'utilisateur (UDF).

SVL_UDF_LOG est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_UDF_LOG. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
query	bigint	ID de requête. Vous pouvez utiliser cet ID pour joindre d'autres vues et tables système.
message	char(4096)	Message généré par la fonction.
créé	timestamp	Heure à laquelle le journal a été créé.
traceback	char(4096)	Si cette valeur est disponibl e, elle fournit un suivi de la pile pour l'UDF. Pour plus d'informations, consultez Suivi dans la bibliothèque standard Python.

Nom de la colonne	Type de données	Description
funcname	character(256)	Nom de la fonction UDF en cours d'exécution.
node	entier	Le nœud sur lequel le message a été généré.
slice	entier	La tranche sur laquelle le message a été généré.
seq	entier	La séquence du message sur la tranche.

Exemples de requêtes

L'exemple suivant montre comment UDFs gérer les erreurs définies par le système. Le premier bloc présente la définition d'une fonction UDF qui renvoie l'inverse d'un argument. Lorsque vous exécutez la fonction et fournissez un argument 0, comme le montre le deuxième bloc, la fonction renvoie une erreur. La troisième instruction lit le message d'erreur enregistré dans SVL_UDF_LOG

```
-- Create a function to find the inverse of a number

CREATE OR REPLACE FUNCTION  f_udf_inv(a int)
  RETURNS float IMMUTABLE
AS $$
   return 1/a
$$ LANGUAGE plpythonu;

-- Run the function with a 0 argument to create an error
Select f_udf_inv(0) from sales;

-- Query SVL_UDF_LOG to view the message

Select query, created, message::varchar
from svl_udf_log;

 query |          created           | message
-------+----------------------------
+-------------------------------------------------------
```

```
  2211 | 2015-08-22 00:11:12.04819  | ZeroDivisionError: long division or modulo by
 zero\nNone
```

L'exemple suivant ajoute un message de consignation et d'avertissement à la fonction UDF afin qu'une opération de division par zéro génère un message d'avertissement au lieu de s'arrêter avec un message d'erreur.

```
-- Create a function to find the inverse of a number and log a warning

CREATE OR REPLACE FUNCTION f_udf_inv_log(a int)
  RETURNS float IMMUTABLE
 AS $$
  import logging
  logger = logging.getLogger() #get root logger
  if a==0:
    logger.warning('You attempted to divide by zero.\nReturning zero instead of error.
\n')
    return 0
  else:
     return 1/a
$$ LANGUAGE plpythonu;
```

L'exemple suivant exécute la fonction, puis interroge SVL_UDF_LOG pour afficher le message.

```

-- Run the function with a 0 argument to trigger the warning
Select f_udf_inv_log(0) from sales;

-- Query SVL_UDF_LOG to view the message

Select query, created, message::varchar
from svl_udf_log;

query |          created           | message
------+----------------------------+---------------------------------
    0 | 2015-08-22 00:11:12.04819  | You attempted to divide by zero.
                                     Returning zero instead of error.
```

SVL_USER_INFO

Vous pouvez récupérer des données sur les utilisateurs de la base de données Amazon Redshift avec la vue SVL_USER_INFO.

SVL_USER_INFO n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
usename	text	Nom d'utilisateur pour le rôle.
usesysid	entier	ID de l'utilisateur.
usecreatedb	boolean	Valeur qui indique si l'utilisateur dispose des autorisations pour créer des bases de données.
usesuper	boolean	Valeur qui indique si l'utilisateur est un super-utilisateur.
usecatupd	boolean	Valeur qui indique si l'utilisateur peut mettre à jour les catalogues système.
useconnlimit	text	Nombre de connexions que l'utilisateur peut ouvrir.
syslogaccess	text	Valeur qui indique si l'utilisateur a accès aux journaux système. Les deux valeurs possibles sont `RESTRICTED` et `UNRESTRICTED`. `RESTRICTED` signifie que les utilisateurs qui ne sont pas des super-utilisateurs peuvent voir leurs propres enregistrements. `UNRESTRICTED` signifie que les utilisateurs qui ne sont pas des super-utilisateurs peuvent voir tous les enregistrements dans les tables et les vues système pour lesquelles ils disposent de privilèges `SELECT`.
last_ddl_ts	timestamp	Horodatage de la dernière instruction de création de langage de définition de données (DDL) exécutée par l'utilisateur.
sessiontimeout	entier	La durée maximale d'inactivité d'une séance avant son expiration (exprimée en secondes). 0 indique qu'aucun délai d'expiration n'a été défini. Pour plus d'informations sur le paramètre de délai d'inactivité

Nom de la colonne	Type de données	Description
		du cluster, consultez Quotas et limites d'Amazon Redshift du guide de gestion des clusters Amazon Redshift.
external_id	text	Identifiant unique de l'utilisateur dans le fournisseur d'identité tiers.

Exemples de requêtes

La commande suivante récupère les informations d'utilisateur à partir de SVL_USER_INFO.

```
SELECT * FROM SVL_USER_INFO;
```

SVL_VACUUM_PERCENTAGE

La vue SVL_VACUUM_PERCENTAGE indique le pourcentage de blocs de données alloués à une table après l'exécution d'une opération VACUUM. Ce pourcentage indique la quantité d'espace disque récupérée. Consultez la commande VACUUM pour plus d'informations sur l'utilitaire VACUUM.

SVL_VACUUM_PERCENTAGE n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Tout ou partie des données de cette table sont également disponibles dans la vue de surveillance SYS SYS_VACUUM_HISTORY. Les données de la vue de surveillance SYS sont formatées pour être plus faciles à utiliser et à comprendre. Nous vous recommandons d'utiliser la vue de surveillance SYS pour vos requêtes.

Colonnes de la table

Nom de la colonne	Type de données	Description
xid	bigint	ID de transaction de l'instruction VACUUM.

Nom de la colonne	Type de données	Description
table_id	entier	ID de table de la table aspirée.
percentage	bigint	Pourcentage de blocs de données après une opération VACUUM (par rapport au nombre de blocs de le table avant l'exécution de l'opération).

Exemple de requête

La requête suivante affiche le pourcentage pour une opération spécifique sur la table 100238 :

```
select * from svl_vacuum_percentage
where table_id=100238 and xid=2200;

xid  | table_id | percentage
-----+----------+------------
1337 |   100238 |         60
(1 row)
```

Après cette opération VACUUM, la table contenait 60 % des blocs d'origine.

Tables catalogue système

Rubriques

- PG_ATTRIBUTE_INFO
- PG_CLASS_INFO
- PG_DATABASE_INFO
- PG_DEFAULT_ACL
- PG_EXTERNAL_SCHEMA
- PG_LIBRARY
- PG_PROC_INFO
- PG_STATISTIC_INDICATOR

- PG_TABLE_DEF
- PG_USER_INFO
- Interrogation des tables catalogue

Les catalogues système stockent les métadonnées du schéma, telles que les informations sur les tables et les colonnes. Les tables catalogue système ont un préfixe PG.

Les tables catalogue PostgreSQL standard sont accessibles aux utilisateurs Amazon Redshift. Pour plus d'informations sur les catalogues système PostgreSQL, consultez Tables système PostgreSQL

PG_ATTRIBUTE_INFO

PG_ATTRIBUTE_INFO est une vue système Amazon Redshift basée sur la table de catalogue PostgreSQL PG_ATTRIBUTE et la table de catalogue interne PG_ATTRIBUTE_ACL. PG_ATTRIBUTE_INFO contient des détails sur les colonnes d'une table ou d'une vue, y compris les listes de contrôle d'accès aux colonnes, le cas échéant.

Colonnes de la table

PG_ATTRIBUTE_INFO affiche la colonne suivante en plus des colonnes de PG_ATTRIBUTE.

Nom de la colonne	Type de données	Description
attacl	aclitem[]	Privilèges d'accès au niveau de la colonne, le cas échéant, qui ont été accordés spécifiquement sur cette colonne.

PG_CLASS_INFO

PG_CLASS_INFO est une vue système Amazon Redshift qui repose sur les tables catalogue PostgreSQL PG_CLASS et PG_CLASS_EXTENDED. PG_CLASS_INFO inclut des détails sur l'heure de création de table et le style de distribution actuel. Pour de plus amples informations, veuillez consulter Distribution des données pour l'optimisation des requêtes.

PG_CLASS_INFO est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

PG_CLASS_INFO montre les colonnes suivantes en plus des colonnes dans PG_CLASS. La colonne `oid` de PG_CLASS est appelée `reloid` dans la table PG_CLASS_INFO.

Nom de la colonne	Type de données	Description
relcreationtime	timestamp	Heure UTC à laquelle la table a été créée.
releffectivediststyle	entier	Le style de distribution d'une table ou, si la table utilise une distribution automatique, le style de distribution actuel attribué par Amazon Redshift.

La colonne RELEFFECTIVEDISTSTYLE de PG_CLASS_INFO indique le style de distribution actuel pour la table. Si la table utilise la distribution automatique, RELEFFECTIVEDISTSTYLE a pour valeur 10, 11 ou 12, ce qui indique si le style de distribution effectif est AUTO (ALL), AUTO (EVEN) ou AUTO (KEY). Si la table utilise la distribution automatique, le style de distribution peut initialement afficher AUTO (ALL), puis passer à AUTO (EVEN) ou AUTO (KEY) lorsque la table se développe ou AUTO (KEY) si une colonne peut être utile en tant que clé de distribution.

La table suivante donne le style de distribution pour chaque valeur de la colonne RELEFFECTIVEDISTSTYLE :

RELEFFECTIVEDISTSTYLE	Style de distribution actuel
0	EVEN
1	KEY
8	ALL
10	AUTO (ALL)
11	AUTO (EVEN)
12	AUTO (KEY)

exemple

La requête suivante renvoie le style de distribution actuel des tables dans le catalogue.

```
select reloid as tableid,trim(nspname) as schemaname,trim(relname) as
 tablename,reldiststyle,releffectivediststyle,
CASE WHEN "reldiststyle" = 0 THEN 'EVEN'::text
     WHEN "reldiststyle" = 1 THEN 'KEY'::text
     WHEN "reldiststyle" = 8 THEN 'ALL'::text
     WHEN "releffectivediststyle" = 10 THEN 'AUTO(ALL)'::text
     WHEN "releffectivediststyle" = 11 THEN 'AUTO(EVEN)'::text
     WHEN "releffectivediststyle" = 12 THEN 'AUTO(KEY)'::text ELSE '<<UNKNOWN>>'::text
 END as diststyle,relcreationtime
from pg_class_info a left join pg_namespace b on a.relnamespace=b.oid;
```

```
 tableid | schemaname | tablename | reldiststyle | releffectivediststyle | diststyle  |
      relcreationtime
---------+------------+-----------+--------------+-----------------------+------------
+----------------------------
 3638033 | public     | customer  |            0 |                     0 | EVEN       |
 2019-06-13 15:02:50.666718
 3638037 | public     | sales     |            1 |                     1 | KEY        |
 2019-06-13 15:03:29.595007
 3638035 | public     | lineitem  |            8 |                     8 | ALL        |
 2019-06-13 15:03:01.378538
 3638039 | public     | product   |            9 |                    10 | AUTO(ALL)  |
 2019-06-13 15:03:42.691611
 3638041 | public     | shipping  |            9 |                    11 | AUTO(EVEN) |
 2019-06-13 15:03:53.69192
 3638043 | public     | support   |            9 |                    12 | AUTO(KEY)  |
 2019-06-13 15:03:59.120695
(6 rows)
```

PG_DATABASE_INFO

PG_DATABASE_INFO est une vue système Amazon Redshift qui étend la table de catalogue PostgreSQL PG_DATABASE.

PG_DATABASE_INFO est visible par tous les utilisateurs.

Colonnes de la table

PG_DATABASE_INFO contient les colonnes suivantes en plus des colonnes de PG_DATABASE. La colonne `oid` de PG_DATABASE est appelée `datid` dans la table PG_DATABASE_INFO. Pour de plus amples informations, veuillez consulter la documentation sur PostgreSQL.

Nom de la colonne	Type de données	Description
datid	oid	Identificateur d'objet utilisé en interne par les tables système.
datconnlimit	text	Nombre maximum de connexions simultanées sur cette base de données. La valeur -1 indique l'absence de limite.

PG_DEFAULT_ACL

Stocke des informations sur les privilèges d'accès par défaut. Pour plus d'informations sur les privilèges d'accès par défaut, consultez ALTER DEFAULT PRIVILEGES.

PG_DEFAULT_ACL est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
defacluser	entier	ID de l'utilisateur auquel les privilèges répertoriés sont appliqués.
defaclnamespace	oid	ID d'objet du schéma auquel les privilèges par défaut sont appliqués. La valeur par défaut est 0 si aucun schéma n'est spécifié.
defaclobjtype	character	Le type d'objet auquel les privilèges par défaut sont appliqués. Les valeurs valides sont les suivantes :

Nom de la colonne	Type de données	Description
		• r — relation (table ou vue) • f — fonction • p — procédure stockée

Nom de la colonne	Type de données	Description
defaclacl	aclitem[]	Chaîne qui définit les privilèges par défaut pour l'utilisateur ou groupe d'utilisateurs et le type d'objet spécifiés. Si les privilèges sont accordés à un utilisateur, la chaîne se présente au format suivant : *{ username=privilegestring/grantor }* nom d’utilisateur Nom de l'utilisateur auquel sont accordés des privilèges. Si username est omis, les privilèges sont accordés à PUBLIC. Si les privilèges sont accordés à un groupe d'utilisateurs, la chaîne se présente au format suivant : *{ "group groupname=privilegestring/grantor" }* privilegestring Chaîne qui spécifie quels privilèges sont accordés. Les valeurs valides sont : • a — INSÉRER (ajouter) • r — SELECT (lire) • w — METTRE À JOUR (écrire) • d — SUPPRIMER • R — RÈGLE • x — Accorde le privilège de créer une contrainte de clé étrangère (REFERENCES). • t — DÉCLENCHEUR • X — EXÉCUTER • U — UTILISATION

Nom de la colonne	Type de données	Description
		• C — CRÉER • T — CRÉER UNE TEMPÉRATURE • D — DÉPOSER • P — TRONQUER • A — MODIFIER • * — Indique que l'utilisateur recevant le privilège précédent peut à son tour accorder le même privilège à d'autres personnes (AVEC L'OPTION GRANT). Une chaîne contenant tous les caractères du code de privilège, classés par position du masque de bits, ressemble à « ArwdRxTxUctDPA ». grantor Nom de l'utilisateur ayant accordé les privilèges. L'exemple suivant indique que l'utilisateur `admin` a accordé tous les privilèges, notamment WITH GRANT OPTION, à l'utilisateur `dbuser`. `dbuser=r*a*w*d*x*X*/admin`

exemple

La requête suivante renvoie tous les privilèges par défaut définis pour la base de données.

```
select pg_get_userbyid(d.defacluser) as user,
n.nspname as schema,
case d.defaclobjtype when 'r' then 'tables' when 'f' then 'functions' end
as object_type,
array_to_string(d.defaclacl, ' + ')  as default_privileges
from pg_catalog.pg_default_acl d
left join pg_catalog.pg_namespace n on n.oid = d.defaclnamespace;
```

```
 user   | schema | object_type |              default_privileges
-------+--------+-------------+-------------------------------------------------------
 admin | tickit | tables      | user1=r/admin + "group group1=a/admin" + user2=w/admin
```

Le résultat de l'exemple précédent indique que pour toutes les nouvelles tables créées par l'utilisateur `admin` dans le schéma `tickit`, `admin` accorde les privilèges SELECT à `user1`, les privilèges INSERT à `group1` et les privilèges UPDATE à `user2`.

PG_EXTERNAL_SCHEMA

Stocke les informations au sujet des schémas externes.

PG_EXTERNAL_SCHEMA est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement les métadonnées auxquelles ils ont accès. Pour plus d'informations, consultez CREATE EXTERNAL SCHEMA.

Colonnes de la table

Nom de la colonne	Type de données	Description
esoid	oid	ID de schéma externe.
eskind	entier	Type de schéma externe.
esdbname	text	Nom de la base de données externe.
esoptions	text	Options de schéma externe.

exemple

L'exemple suivant illustre les détails relatifs aux schémas externes.

```
select esoid, nspname as schemaname, nspowner, esdbname as external_db, esoptions
from pg_namespace a,pg_external_schema b where a.oid=b.esoid;

esoid  | schemaname      | nspowner | external_db | esoptions
```

```
-------+----------------+----------+-------------
+-------------------------------------------------------------
100134 | spectrum_schema |      100 | spectrum_db |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
100135 | spectrum        |      100 | spectrumdb  |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
100149 | external        |      100 | external_db |
 {"IAM_ROLE":"arn:aws:iam::123456789012:role/mySpectrumRole"}
```

PG_LIBRARY

Stocke les informations sur les bibliothèques définies par l'utilisateur.

PG_LIBRARY est visible par tous les utilisateurs. Les super-utilisateurs peuvent voir toutes les lignes, tandis que les utilisateurs standard peuvent voir uniquement leurs propres données. Pour plus d'informations, consultez Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
name	name	Nom de la bibliothèque.
language_oid	oid	Réservé au système.
file_store_id	entier	Réservé au système.
owner	entier	ID d'utilisateur du propriétaire de la bibliothèque.

exemple

L'exemple suivant renvoie des informations pour les bibliothèques installées par l'utilisateur.

```
select * from pg_library;

name       | language_oid | file_store_id | owner
```

```
-----------+--------------+---------------+------
f_urlparse |       108254 |          2000 |   100
```

PG_PROC_INFO

PG_PROC_INFO est une vue système Amazon Redshift basée sur la table de catalogue PostgreSQL PG_PROC et la table de catalogue interne PG_PROC_EXTENDED. PG_PROC_INFO comprend des détails sur les procédures stockées et les fonctions, notamment des informations relatives aux arguments en sortie, le cas échéant.

Colonnes de la table

PG_PROC_INFO affiche les colonnes suivantes en plus des colonnes dans PG_PROC. La colonne `oid` de PG_PROC est appelée `prooid` dans la table PG_PROC_INFO.

Nom de la colonne	Type de données	Description
prooid	oid	ID d'objet de la fonction ou de la procédure stockée.
prokind	"char"	Valeur indiquant le type de fonctions ou de procédures stockées. La valeur est « f » pour les fonctions régulières, « p » pour les procédures stockées et « a » pour les fonctions d'agrégation.
proargmodes	"char"[ ]	Tableau contenant les modes des arguments de la procédure, codés sous la forme « i » pour les arguments IN, « o » pour les arguments OUT et « b » pour les arguments INOUT. Si tous les arguments sont des arguments IN, ce champ est NULL. Les indices correspondent à des positions dans le tableau proallargtypes.
proallargtypes	oid[ ]	Tableau contenant les types de données des arguments de la procédure. Ce tableau contient tous les types d'arguments (y compris les arguments OUT et INOUT). Cependant, si tous les arguments sont des arguments IN, ce champ est NULL. L'indexation est basée sur un. Les indices des valeurs proargtypes, en revanche, commencent à 0.

Le champ proargnames dans PG_PROC_INFO contient les noms de tous les types d'arguments (y compris OUT et INOUT), le cas échéant.

PG_STATISTIC_INDICATOR

Stocke les informations sur le nombre de lignes insérées ou supprimées depuis la dernière opération ANALYZE. La table PG_STATISTIC_INDICATOR est mise à jour régulièrement en fonction des opérations DML ; les statistiques sont donc approximatives.

PG_STATISTIC_INDICATOR n'est visible que par les super-utilisateurs. Pour de plus amples informations, veuillez consulter Visibilité des données dans les tables et vues système.

Colonnes de la table

Nom de la colonne	Type de données	Description
stairelid	oid	ID de table
stairows	float	Nombre total de lignes de la table.
staiins	float	Nombre de lignes insérées depuis la dernière opération ANALYZE.
staidels	float	Nombre de lignes supprimées ou mises à jour depuis la dernière opération ANALYZE.

exemple

L'exemple suivant renvoie les informations sur les modifications apportées à la table depuis la dernière opération ANALYZE.

```
select * from pg_statistic_indicator;

stairelid | stairows | staiins | staidels
----------+----------+---------+---------
   108271 |       11 |       0 |        0
   108275 |      365 |       0 |        0
   108278 |     8798 |       0 |        0
   108280 |    91865 |       0 |   100632
```

```
 108267 |    89981 |   49990 |     9999
 108269 |      808 |     606 |      374
 108282 |   152220 |   76110 |   248566
```

PG_TABLE_DEF

Stocke les informations sur les colonnes de la table.

PG_TABLE_DEF renvoie uniquement les informations sur les tables visibles de l'utilisateur. Si PG_TABLE_DEF ne renvoie pas les résultats attendus, vérifiez que le paramètre search_path est correctement défini pour inclure les schémas correspondants.

Vous pouvez utiliser SVV_TABLE_INFO pour afficher des informations plus exhaustives sur une table, notamment l'asymétrie de la distribution des données, l'asymétrie de la distribution de clés, la taille de la table et les statistiques.

Colonnes de la table

Nom de la colonne	Type de données	Description
schemaname	name	Nom du schéma.
tablename	name	Nom de la table.
column	name	Nom de la colonne.
type	text	Type de données de la colonne.
encoding	character(32)	Encodage de la colonne.
distkey	boolean	True si cette colonne est la clé de distribution de la table.
sortkey	entier	Ordre de la colonne dans la clé de tri. Si la table utilise une clé de tri composée et que toutes les colonnes qui font partie de la clé de tri ont une valeur positive qui indique la position de la colonne dans la clé de tri. Si la table utilise une clé de tri entrelacée, toutes les colonnes qui font partie de la clé de tri ont une valeur qui est tour à tour positive

Nom de la colonne	Type de données	Description
		ou négative, où la valeur absolue indique la position de la colonne dans la clé de tri. Si 0, la colonne de fait pas partie d'une clé de tri.
notnull	boolean	True si la colonne comporte une contrainte NOT NULL.

exemple

L'exemple suivant illustre les colonnes de clés de tri composées de la table LINEORDER_COMPOUND.

```
select "column", type, encoding, distkey, sortkey, "notnull"
from pg_table_def
where tablename = 'lineorder_compound'
and sortkey <> 0;

column       | type    | encoding | distkey | sortkey | notnull
-------------+---------+----------+---------+---------+--------
lo_orderkey  | integer | delta32k | false   |       1 | true
lo_custkey   | integer | none     | false   |       2 | true
lo_partkey   | integer | none     | true    |       3 | true
lo_suppkey   | integer | delta32k | false   |       4 | true
lo_orderdate | integer | delta    | false   |       5 | true
(5 rows)
```

L'exemple suivant illustre les colonnes de clés de tri entrelacées de la table LINEORDER_INTERLEAVED.

```
select "column", type, encoding, distkey, sortkey, "notnull"
from pg_table_def
where tablename = 'lineorder_interleaved'
and sortkey <> 0;

column       | type    | encoding | distkey | sortkey | notnull
-------------+---------+----------+---------+---------+--------
lo_orderkey  | integer | delta32k | false   |      -1 | true
lo_custkey   | integer | none     | false   |       2 | true
lo_partkey   | integer | none     | true    |      -3 | true
```

```
lo_suppkey   | integer | delta32k | false   |       4 | true
lo_orderdate | integer | delta    | false   |      -5 | true
(5 rows)
```

PG_TABLE_DEF renvoie uniquement les informations pour les tables dans des schémas inclus dans le chemin de recherche. Pour de plus amples informations, veuillez consulter search_path.

Par exemple, supposons que vous créiez un schéma et une table, puis que vous interrogiez PG_TABLE_DEF.

```
create schema demo;
create table demo.demotable (one int);
select * from pg_table_def where tablename = 'demotable';

schemaname|tablename|column| type | encoding | distkey | sortkey | notnull
----------+---------+------+------+----------+---------+---------+--------
```

La requête ne renvoie aucune ligne pour la nouvelle table. Examinez le paramètre de `search_path`.

```
show search_path;

  search_path
---------------
 $user, public
(1 row)
```

Ajoutez le schéma demo au chemin de recherche et exécutez la requête à nouveau.

```
set search_path to '$user', 'public', 'demo';

select * from pg_table_def where tablename = 'demotable';

schemaname| tablename |column|  type   | encoding |distkey|sortkey| notnull
----------+-----------+------+---------+----------+-------+-------+--------
demo      | demotable | one  | integer | none     | f     |     0 | f
(1 row)
```

PG_USER_INFO

PG_USER_INFO est une vue système d'Amazon Redshift qui affiche des informations sur les utilisateurs, telles que l'ID utilisateur et le délai d'expiration du mot de passe.

Seuls les super-utilisateurs peuvent afficher PG_USER_INFO.

Colonnes de la table

PG_USER_INFO contient les colonnes suivantes. Pour de plus amples informations, veuillez consulter la documentation sur PostgreSQL.

Nom de la colonne	Type de données	Description
usename	name	Le nom d'utilisateur.
usesysid	entier	L'ID utilisateur.
usecreatedb	boolean	True (Vrai) si l'utilisateur peut créer des bases de données.
usesuper	boolean	True (Vrai) si l'utilisateur est un super-utilisateur.
usecatupd	boolean	True (Vrai) si l'utilisateur peut mettre à jour les catalogues système.
passwd	text	Le mot de passe.
valuntil	abstime	La date et l'heure d'expiration du mot de passe.
useconfig	text[]	Les valeurs par défaut de la session pour les variables d'exécution.
useconnlimit	text	Nombre de connexions que l’utilisateur peut ouvrir.

Interrogation des tables catalogue

Rubriques

- Exemples de requêtes de catalogue

En général, vous pouvez joindre des tables et des vues du catalogue (relations dont les noms commencent par **PG_**) à des tables et des vues Amazon Redshift.

Les tables du catalogue utilisent un certain nombre de types de données non pris en charge par Amazon Redshift. Les types de données suivants sont pris en charge lorsque des requêtes joignent des tables de catalogue à des tables Amazon Redshift :

- bool
- "char"
- float4
- int2
- int4
- int8
- name
- oid
- text
- varchar

Si vous écrivez une requête de jointure qui fait référence explicitement ou implicitement à une colonne avec un type de données non pris en charge, la requête renvoie une erreur. Les fonctions SQL qui sont utilisées dans certaines des tables catalogue ne sont également pas prises en charge, à l'exception de celles utilisées par les tables PG_SETTINGS et PG_LOCKS.

Par exemple, la table PG_STATS ne peut pas être interrogée dans une jointure avec une table Amazon Redshift en raison de fonctions non prises en charge.

Les tables et vues du catalogue suivantes fournissent des informations utiles qui peuvent être jointes à des informations dans les tables Amazon Redshift. Certaines de ces tables autorisent uniquement un accès partiel en raison de restrictions de type de données et de fonction. Lorsque vous interrogez les tables partiellement accessibles, sélectionnez ou faites référence à leurs colonnes avec soin.

Les tables suivantes sont totalement accessibles et ne contiennent pas les types ou fonctions non pris en charge :

- pg_attribute
- pg_cast
- pg_depend
- pg_description

- pg_locks
- pg_opclass

Les tables suivantes sont partiellement accessibles et contiennent certains types, fonctions et colonnes de texte tronquées non pris en charge. Les valeurs dans les colonnes de texte sont tronquées en valeurs varchar(256).

- pg_class
- pg_constraint
- pg_database
- pg_group
- pg_language
- pg_namespace
- pg_operator
- pg_proc
- pg_settings
- pg_statistic
- pg_tables
- pg_type
- pg_user
- pg_views

Les tables du catalogue qui ne sont pas répertoriées ici sont inaccessibles ou probablement inutiles pour les administrateurs Amazon Redshift. Cependant, vous pouvez interroger librement une table ou une vue du catalogue si votre requête n'implique pas de jointure avec une table Amazon Redshift.

Vous pouvez utiliser les colonnes OID dans les tables catalogue Postgres en tant que colonnes de jointure. Par exemple, la condition de jointure `pg_database.oid = stv_tbl_perm.db_id` correspond à l'ID d'objet de base de données interne de chaque ligne PG_DATABASE avec la colonne DB_ID visible dans la table STV_TBL_PERM. Les colonnes OID sont des clés primaires internes qui ne sont pas visibles lorsque vous effectuez la sélection depuis la table. Les vues catalogue ne disposent pas de colonnes OID.

Certaines fonctions Amazon Redshift ne doivent s'exécuter que sur les nœuds de calcul. Si une requête fait référence à une table créée par un utilisateur, l'instruction SQL s'exécute sur les nœuds de calcul.

Une requête qui ne fait référence qu'aux tables catalogue (tables avec le préfixe PG, comme PG_TABLE_DEF) ou qui ne fait référence à aucune table, s'exécute exclusivement sur le nœud principal.

Si une requête qui utilise une fonction de nœud de calcul ne fait pas référence à une table définie par un utilisateur ou à une table système Amazon Redshift, elle renvoie l'erreur suivante :

```
[Amazon](500310) Invalid operation: One or more of the used functions must be applied
 on at least one user created table.
```

Exemples de requêtes de catalogue

Les requêtes suivantes illustrent quelques-unes des manières dont vous pouvez interroger les tables du catalogue pour obtenir des informations utiles sur une base de données Amazon Redshift.

Afficher l'ID de la table, la base de données, le schéma et le nom de la table

La définition de vue suivante joint la table système STV_TBL_PERM aux tables catalogue système PG_CLASS, PG_NAMESPACE et PG_DATABASE pour renvoyer l'ID de la table, le nom de la base de données, le nom du schéma et le nom de la table.

```
create view tables_vw as
select distinct(stv_tbl_perm.id) table_id
,trim(pg_database.datname)   db_name
,trim(pg_namespace.nspname)   schema_name
,trim(pg_class.relname)   table_name
from stv_tbl_perm
join pg_class on pg_class.oid = stv_tbl_perm.id
join pg_namespace on pg_namespace.oid = pg_class.relnamespace
join pg_database on pg_database.oid = stv_tbl_perm.db_id;
```

L'exemple suivant renvoie les informations concernant l'ID de table 117855.

```
select * from tables_vw where table_id = 117855;
```

```
table_id | db_name   | schema_name | table_name
```

```
---------+-----------+-------------+-----------
  117855 |       dev | public      | customer
```

Répertorier le nombre de colonnes par table Amazon Redshift

La requête suivante joint certaines tables de catalogue pour connaître le nombre de colonnes que contient chaque table Amazon Redshift. Les noms de table Amazon Redshift sont stockés à la fois dans PG_TABLES et STV_TBL_PERM ; lorsque cela est possible, utilisez PG_TABLES pour renvoyer les noms de table Amazon Redshift.

Cette requête n'implique pas de table Amazon Redshift.

```
select nspname, relname, max(attnum) as num_cols
from pg_attribute a, pg_namespace n, pg_class c
where n.oid = c.relnamespace and  a.attrelid = c.oid
and c.relname not like '%pkey'
and n.nspname not like 'pg%'
and n.nspname not like 'information%'
group by 1, 2
order by 1, 2;

nspname | relname  | num_cols
--------+----------+----------
public  | category |        4
public  | date     |        8
public  | event    |        6
public  | listing  |        8
public  | sales    |       10
public  | users    |       18
public  | venue    |        5
(7 rows)
```

Répertorier les schémas et les tables dans une base de données

La requête suivante joint STV_TBL_PERM à quelques tables PG pour renvoyer une liste de tables de la base de données TICKIT et leurs noms de schéma (colonne NSPNAME). La requête renvoie également le nombre total de lignes de chaque table. (Cette requête est utile lorsque plusieurs schémas de votre système portent des noms de tables identiques.)

```
select datname, nspname, relname, sum(rows) as rows
```

```
from pg_class, pg_namespace, pg_database, stv_tbl_perm
where pg_namespace.oid = relnamespace
and pg_class.oid = stv_tbl_perm.id
and pg_database.oid = stv_tbl_perm.db_id
and datname ='tickit'
group by datname, nspname, relname
order by datname, nspname, relname;

datname | nspname | relname  |  rows
--------+---------+----------+--------
tickit  | public  | category |     11
tickit  | public  | date     |    365
tickit  | public  | event    |   8798
tickit  | public  | listing  | 192497
tickit  | public  | sales    | 172456
tickit  | public  | users    |  49990
tickit  | public  | venue    |    202
(7 rows)
```

Table de liste IDs, types de données, noms de colonnes et noms de tables

La requête suivante répertorie quelques informations sur chaque table d'utilisateur et ses colonnes : l'ID de la table, le nom de la table, les noms de ses colonnes et le type de données de chaque colonne :

```
select distinct attrelid, rtrim(name), attname, typname
from pg_attribute a, pg_type t, stv_tbl_perm p
where t.oid=a.atttypid and a.attrelid=p.id
and a.attrelid between 100100 and 110000
and typname not in('oid','xid','tid','cid')
order by a.attrelid asc, typname, attname;

attrelid |  rtrim   |    attname     |  typname
---------+----------+----------------+-----------
  100133 | users    | likebroadway   | bool
  100133 | users    | likeclassical  | bool
  100133 | users    | likeconcerts   | bool
...
  100137 | venue    | venuestate     | bpchar
  100137 | venue    | venueid        | int2
  100137 | venue    | venueseats     | int4
  100137 | venue    | venuecity      | varchar
...
```

Compter le nombre de blocs de données de chaque colonne dans une table

La requête suivante joint la table STV_BLOCKLIST à PG_CLASS pour renvoyer des informations de stockage sur les colonnes de la table SALES.

```
select col, count(*)
from stv_blocklist s, pg_class p
where s.tbl=p.oid and relname='sales'
group by col
order by col;

col | count
----+-------
  0 |     4
  1 |     4
  2 |     4
  3 |     4
  4 |     4
  5 |     4
  6 |     4
  7 |     4
  8 |     4
  9 |     8
 10 |     4
 12 |     4
 13 |     8
(13 rows)
```

Référence de configuration

Vous pouvez utiliser des configurations pour personnaliser votre environnement. Avec Amazon Redshift, vous pouvez personnaliser et optimiser votre environnement d'entreposage de données en configurant différents paramètres et réglages. La référence de configuration décrit les propriétés du cluster, les paramètres de base de données et les options de configuration de gestion de la charge de travail (WLM) disponibles. Vous pouvez consulter cette référence pour affiner les performances, la sécurité et l'allocation des ressources en fonction de leurs besoins spécifiques. La référence suivante fournit des conseils détaillés sur la modification de ces configurations pour obtenir la configuration d'entrepôt de données souhaitée.

Rubriques

- Modification de la configuration du serveur
- analyze_threshold_percent
- cast_super_null_on_error
- datashare_break_glass_session_var
- datestyle
- default_geometry_encoding
- describe_field_name_in_uppercase
- downcase_delimited_identificateur
- enable_case_sensitive_identifier
- enable_case_sensitive_super_attribute
- enable_numeric_rounding
- enable_result_cache_for_session
- enable_vacuum_boost
- error_on_nondeterministic_update
- extra_float_digits
- interval_forbid_composite_literals
- json_serialization_enable
- json_serialization_parse_nested_strings
- max_concurrency_scaling_clusters
- max_cursor_result_set_size

- mv_enable_aqmv_for_session
- navigate_super_null_on_error
- parse_super_null_on_error
- pg_federation_repeatable_read
- query_group
- search_path
- spectrum_enable_pseudo_columns
- enable_spectrum_oid
- spectrum_query_maxerror
- statement_timeout
- stored_proc_log_min_messages
- timezone
- use_fips_ssl
- wlm_query_slot_count

Modification de la configuration du serveur

Vous pouvez modifier la configuration du serveur de différentes façons :

- Avec une commande SET pour remplacer un paramètre pendant la durée de la séance actuelle uniquement.

  Par exemple :

  ```
  set extra_float_digits to 2;
  ```

- En modifiant les valeurs du groupe de paramètres pour le cluster. Les valeurs du groupe de paramètres incluent les paramètres supplémentaires que vous pouvez configurer. Pour plus d'informations, consultez la rubrique Groupes de paramètres Amazon Redshift dans le Guide de gestion Amazon Redshift.
- En utilisant la commande ALTER USER pour définir un paramètre de configuration avec une nouvelle valeur pour toutes les séances exécutées par l'utilisateur spécifié.

  ```
  ALTER USER username SET parameter { TO | = } { value | DEFAULT }
  ```

Utilisez la commande SHOW pour afficher les valeurs de paramètre en cours. Utilisez SHOW ALL pour afficher tous les paramètres que vous pouvez configurer à l'aide de la commande SET.

```
SHOW ALL;

name                        | setting
----------------------------+--------------
analyze_threshold_percent | 10
datestyle                   | ISO, MDY
extra_float_digits          | 2
query_group                 | default
search_path                 | $user, public
statement_timeout           | 0
timezone                    | UTC
wlm_query_slot_count        | 1
```

> **Note**
>
> Notez que les paramètres de configuration sont appliqués à la base de données à laquelle vous êtes connecté dans votre entrepôt de données.

analyze_threshold_percent

Valeurs (par défaut en gras)

10, 0 à 100.0

Description

Définit le seuil du pourcentage de lignes modifiées pour l'analyse d'une table. Pour réduire le délai de traitement et améliorer les performances globales du système, Amazon Redshift ignore les opérations ANALYZE pour les tables dont le pourcentage de lignes modifiées est inférieur à celui spécifié par `analyze_threshold_percent`. Par exemple, si une table contient 100 000 000 lignes et que 9 000 000 lignes ont été modifiées depuis la dernière opération ANALYZE, la table est ignorée par défaut, car moins de 10 % des lignes ont été modifiées. Pour analyser des tables lorsque seul un petit nombre de lignes a changé, définissez `analyze_threshold_percent` sur un petit nombre arbitraire. Par exemple, si vous définissez `analyze_threshold_percent` sur 0,01, une table avec 100 000 000 lignes ne sera pas ignorée si au moins 10 000 lignes ont été modifiées. Pour analyser

toutes les tables même si aucune ligne n'a été modifiée, définissez `analyze_threshold_percent` sur 0.

Vous pouvez modifier le paramètre `analyze_threshold_percent` pour la séance actuelle seulement en utilisant une commande SET. Le paramètre ne peut pas être modifié dans un groupe de paramètres.

exemple

```
set analyze_threshold_percent to 15;
set analyze_threshold_percent to 0.01;
set analyze_threshold_percent to 0;
```

cast_super_null_on_error

Valeurs (par défaut en gras)

on, off

Description

Spécifie que lorsque vous essayez d'accéder à un membre inexistant d'un objet ou d'un élément d'un tableau, Amazon Redshift renvoie une valeur NULL si votre requête est exécutée dans le mode par défaut « lax ».

datashare_break_glass_session_var

Valeurs (par défaut en gras)

Il n'existe aucune valeur par défaut. La valeur peut être n'importe quelle chaîne de caractères générée par Amazon Redshift lorsqu'une opération n'est pas recommandée, comme décrit ci-dessous.

Description

Applique une autorisation autorisant certaines opérations qui ne sont généralement pas recommandées pour un partage de AWS Data Exchange données.

En général, nous vous recommandons de ne pas supprimer ou modifier un partage de données à l'aide de l'instruction DROP AWS Data Exchange DATASHARE ou ALTER DATASHARE SET PUBLICACCESSIBLE. Pour autoriser la suppression ou la modification d'un AWS Data Exchange partage de données afin de désactiver le paramètre accessible au public, définissez la `datashare_break_glass_session_var` variable sur une valeur unique. Cette valeur ponctuelle est générée par Amazon Redshift et fournie dans un message d'erreur après la tentative initiale de l'opération en question.

Après avoir défini la variable sur la valeur générée unique, exécutez à nouveau l'instruction DROP DATASHARE ou ALTER DATASHARE.

Pour plus d'informations, consultez Notes d'utilisation d'ALTER DATASHARE ou Note d'utilisation de DROP DATASHARE.

exemple

```
set datashare_break_glass_session_var to '620c871f890c49';
```

datestyle

Valeurs (par défaut en gras)

Spécification du format (ISO, Postgres, SQL ou allemand) et year/month/day classement (DMY, MDY, YMD).

- ISO — utilise le style de date YYYY-MM-DD HH:MM:SS.
- Postgres – utilise le style de date MM-JJ HH:MM:SS AAAA.
- SQL — utilise le style de date MM-DD-YYYY HH:MM:SS.
- Allemand — utilise le style de date DD-MM-YYYY HH:MM:SS.

Description

Définit le format d'affichage pour les valeurs de date et heure, ainsi que les règles d'interprétation des valeurs d'entrée de date ambiguës. La chaîne contient deux paramètres qui peuvent être modifiés séparément ou ensemble.

exemple

```
show datestyle;
DateStyle
-----------
ISO, MDY
(1 row)

set datestyle to 'SQL,DMY';
```

default_geometry_encoding

Valeurs (par défaut en gras)

1, 2

Description

Configuration de séance qui spécifie si les géométries spatiales créées au cours de cette séance sont encodées avec un cadre de sélection. Si `default_geometry_encoding` est 1, les géométries ne sont pas encodées avec un cadre de sélection. Si `default_geometry_encoding` est 2, les géométries sont encodées avec un cadre de sélection. Pour plus d'informations sur la prise en charge des cadres de sélection, consultez Cadre de délimitation.

describe_field_name_in_uppercase

Valeurs (par défaut en gras)

off (faux), on (vrai)

Description

Spécifie si les noms de colonnes retournés par les instructions SELECT sont en majuscules ou en minuscules. Si ce paramètre est « on », les noms de colonnes sont renvoyés en majuscules. Si ce paramètre est « off », les noms de colonnes sont renvoyés en minuscules. Amazon Redshift stocke les noms de colonnes en minuscules, quelle que soit la valeur de `describe_field_name_in_uppercase`.

exemple

```
set describe_field_name_in_uppercase to on;

show describe_field_name_in_uppercase;

DESCRIBE_FIELD_NAME_IN_UPPERCASE
--------------------------------
on
```

downcase_delimited_identificateur

Valeurs (par défaut en gras)

on, off

Description

La mise hors service de cette configuration est en cours. Utilisez plutôt `enable_case_sensitive_identifier`.

Permet au super parseur de lire les champs JSON en majuscules ou en casse mixte. Permet également la prise en charge des requêtes fédérées pour les bases de données PostgreSQL prises en charge avec des noms de base de données, de schéma, de table et de colonne en casse mixte. Pour utiliser des identifiants sensibles à la casse, réglez ce paramètre sur « off ».

Notes d'utilisation

- Si vous utilisez des fonctionnalités de sécurité au niveau des lignes ou de masquage des données dynamiques, nous vous recommandons de définir la valeur `downcase_delimited_identifier` dans le groupe de paramètres de votre cluster ou groupe de travail. Cela garantit que `downcase_delimited_identifier` reste constant tout au long de la création et de l'attachement d'une politique, puis de l'interrogation d'une relation à laquelle une politique est appliquée. Pour plus d'informations sur la sécurité au niveau des lignes, consultez Sécurité au niveau des lignes. Pour plus d'informations sur le masquage des données dynamiques, consultez Masquage dynamique des données.
- Lorsque vous définissez `downcase_delimited_identifier` sur off et que vous créez une table, vous pouvez définir des noms de colonne sensibles à la casse. Lorsque vous définissez

downcase_delimited_identifier sur on et que vous interrogez la table, les noms des colonnes sont en minuscules. Cela peut produire des résultats de requête différents de ceux obtenus lorsque downcase_delimited_identifier est défini sur off. Prenez l'exemple suivant :

```
SET downcase_delimited_identifier TO off;
--Amazon Redshift preserves case for column names and other identifiers.

--Create a table with two columns that are identical except for the case.
CREATE TABLE t ("c" int, "C" int);

INSERT INTO t VALUES (1, 2);

SELECT * FROM t;

 c | C
---+---
 1 | 2
(1 row)

SET enable_downcase_delimited_identifier TO on;
--Amazon Redshift no longer preserves case for column names and other identifiers.

SELECT * FROM t;

 c | c
---+---
 1 | 1
(1 row)
```

- Nous recommandons aux utilisateurs standard interrogeant des tables associées à un masquage dynamique des données ou à des politiques de sécurité au niveau des lignes d'utiliser le paramètre downcase_delimited_identifier par défaut. Pour obtenir des informations sur la sécurité au niveau des lignes, consultez Sécurité au niveau des lignes. Pour plus d'informations sur le masquage dynamique des données, consultez Masquage dynamique des données.

enable_case_sensitive_identifier

Valeurs (par défaut en gras)

true, false

Description

Valeur de configuration qui détermine si les identificateurs de nom des bases de données, des schémas, des tables et des colonnes distinguent les majuscules et minuscules. Le cas des identifiants de nom est préservé lorsque vous les placez entre guillemets doubles et que vous les définissez sur. `enable_case_sensitive_identifier` `true` Le cas des identificateurs de nom n'est pas conservé, mais converti en minuscules, lorsque vous ne placez pas les identifiants entre guillemets ou lorsque vous définissez sur. `enable_case_sensitive_identifier` `false`

La casse d'un nom d'utilisateur entre guillemets est toujours préservée, quel que soit le paramètre de l'option de configuration `enable_case_sensitive_identifier`.

Exemples

L'exemple suivant montre comment créer et utiliser des identifiants sensibles à la casse pour le nom de la table et de la colonne.

```
-- To create and use case sensitive identifiers
SET enable_case_sensitive_identifier TO true;

-- Create tables and columns with case sensitive identifiers
CREATE TABLE public."MixedCasedTable" ("MixedCasedColumn" int);

INSERT INTO public."MixedCasedTable" VALUES (1);
INSERT INTO public."MixedCasedTable" VALUES (2);
INSERT INTO public."MixedCasedTable" VALUES (3);
INSERT INTO public."MixedCasedTable" VALUES (4);
INSERT INTO public."MixedCasedTable" VALUES (5);

-- Now query with case sensitive identifiers
SELECT "MixedCasedColumn" FROM public."MixedCasedTable";

MixedCasedColumn
------------------
1
2
3
4
5

(5 rows)
```

```
SELECT * FROM public."MixedCasedTable" WHERE "MixedCasedColumn" = 1;

mixedcasedcolumn
------------------
1

(1 row)
```

L'exemple suivant montre quand la casse des identifiants n'est pas conservée.

```
-- To not use case sensitive identifiers
RESET enable_case_sensitive_identifier;

-- Mixed case identifiers are lowercased despite double quotation marks
CREATE TABLE "MixedCasedTable2" ("MixedCasedColumn" int);

CREATE TABLE MixedCasedTable2 (MixedCasedColumn int);

ERROR:  Relation "mixedcasedtable2" already exists


SELECT "MixedCasedColumn" FROM "MixedCasedTable2";

 mixedcasedcolumn
------------------
(0 rows)


SELECT MixedCasedColumn FROM MixedCasedTable2;

 mixedcasedcolumn
------------------
(0 rows)
```

Notes d’utilisation

- Si vous utilisez l'actualisation automatique pour les vues matérialisées, nous vous recommandons de définir la valeur `enable_case_sensitive_identifier` dans le groupe de paramètres de votre cluster ou de votre groupe de travail. Cela permet de garantir que `enable_case_sensitive_identifier` reste constant lorsque vos vues matérialisées sont

actualisées. Pour plus d'informations sur l'actualisation automatique des vues matérialisées, consultez Actualisation d'une vue matérialisée. Pour plus d'informations sur la définition des valeurs de configuration dans les groupes de paramètres, reportez-vous à Groupes de paramètres Amazon Redshift dans le Guide de gestion Amazon Redshift.

- Si vous utilisez des fonctionnalités de sécurité au niveau des lignes ou de masquage des données dynamiques, nous vous recommandons de définir la valeur `enable_case_sensitive_identifier` dans le groupe de paramètres de votre cluster ou groupe de travail. Cela garantit que `enable_case_sensitive_identifier` reste constant tout au long de la création et de l'attachement d'une politique, puis de l'interrogation d'une relation à laquelle une politique est appliquée. Pour plus d'informations sur la sécurité au niveau des lignes, consultez Sécurité au niveau des lignes. Pour plus d'informations sur le masquage des données dynamiques, consultez Masquage dynamique des données.
- Lorsque vous définissez `enable_case_sensitive_identifier` sur on et que vous créez une table, vous pouvez définir des noms de colonne sensibles à la casse. Lorsque vous définissez `enable_case_sensitive_identifier` sur off et que vous interrogez la table, les noms des colonnes sont en minuscules. Cela peut produire des résultats de requête différents de ceux obtenus lorsque `enable_case_sensitive_identifier` est défini sur on. Prenez l'exemple suivant :

```
SET enable_case_sensitive_identifier TO on;
--Amazon Redshift preserves case for column names and other identifiers.

--Create a table with two columns that are identical except for the case.
CREATE TABLE t ("c" int, "C" int);

INSERT INTO t VALUES (1, 2);

SELECT * FROM t;

 c | C
---+---
 1 | 2
(1 row)

SET enable_case_sensitive_identifier TO off;
--Amazon Redshift no longer preserves case for column names and other identifiers.

SELECT * FROM t;

 c | c
```

```
---+---
 1 | 1
(1 row)
```

- Nous recommandons aux utilisateurs standard interrogeant des tables associées à un masquage dynamique des données ou à des politiques de sécurité au niveau des lignes d'utiliser le paramètre enable_case_sensitive_identifier par défaut. Pour plus d'informations sur la sécurité au niveau des lignes, consultez Sécurité au niveau des lignes. Pour plus d'informations sur le masquage dynamique des données, consultez Masquage dynamique des données.
- Pour référencer des identificateurs comportant une combinaison de majuscules et minuscules à l'aide de la notation par points, placez chaque identifiant entre guillemets doubles. Par exemple, `public."MixedCasedTable"."MixedCasedColumn"`.

enable_case_sensitive_super_attribute

Valeurs (par défaut en gras)

true, false

Description

Valeur de configuration qui détermine si la navigation dans les structures de type de données SUPER avec des noms d'attributs non délimités est sensible à la casse. Lorsque vous définissez `enable_case_sensitive_super_attribute` sur `true`, la navigation dans les structures de type SUPER avec des noms d'attributs non délimités est sensible à la casse. Lorsque vous définissez la valeur sur `false`, la navigation dans les structures de type SUPER avec des noms d'attributs non délimités n'est pas sensible à la casse.

Lorsque vous mettez un nom d'attribut entre guillemets doubles et que vous définissez `enable_case_sensitive_identifier` sur `true`, la casse est toujours préservée, quel que soit le réglage de l'option de configuration `enable_case_sensitive_super_attribute`.

`enable_case_sensitive_super_attribute` ne s'applique qu'aux colonnes ayant le type de données SUPER. Pour toutes les autres colonnes, envisagez d'utiliser `enable_case_sensitive_identifier` à la place

Pour plus d'informations sur le type de données SUPER, consultez Type SUPER et Données semi-structurées dans Amazon Redshift.

Exemples

L'exemple suivant montre les résultats de la sélection des valeurs SUPER avec `enable_case_sensitive_super_attribute` activé et désactivé.

```
--Create a table with a SUPER column.
CREATE TABLE tbl (col SUPER);

--Insert values.
INSERT INTO tbl VALUES (json_parse('{
 "A": "HELLO", "a": "123"
}'));

SET enable_case_sensitive_super_attribute TO ON;

SELECT col.A FROM tbl;
  a
-----
 "HELLO"
(1 row)

SELECT col.a FROM tbl;
  a
-----
 "123"
(1 row)

SET enable_case_sensitive_super_attribute TO OFF;

SELECT col.A FROM tbl;
  a
-----
 "123"
(1 row)

SELECT col.a FROM tbl;
  a
-----
 "123"
(1 row)
```

Notes d’utilisation

- Les vues et les vues matérialisées suivent la valeur de `enable_case_sensitive_super_attribute` au moment de leur création. Les vues, procédures stockées et fonctions définies par l'utilisateur liées tardivement suivent la valeur de `enable_case_sensitive_super_attribute` au moment de l'interrogation.
- Si vous utilisez l'actualisation automatique pour les vues matérialisées, nous vous recommandons de définir le paramètre `enable_case_sensitive_identifier value` dans le groupe de paramètres de votre cluster ou de votre groupe de travail. Cela permet de garantir que `enable_case_sensitive_identifier` reste constant lorsque vos vues matérialisées sont actualisées. Pour plus d'informations sur l'actualisation automatique des vues matérialisées, consultez Actualisation d’une vue matérialisée. Pour plus d'informations sur la définition des valeurs de configuration dans les groupes de paramètres, reportez-vous à Groupes de paramètres Amazon Redshift dans le Guide de gestion Amazon Redshift.
- Le nom de colonne dans les résultats de l'instruction est toujours mis en minuscules, quelle que soit la valeur de `enable_case_sensitive_super_attribute`. Pour que le nom de la colonne soit également sensible à la casse, activez `enable_case_sensitive_identifier`.
- Nous recommandons aux utilisateurs standard interrogeant des tables associées à des politiques de sécurité au niveau des lignes d'utiliser le paramètre `enable_case_sensitive_identifier` par défaut. Pour obtenir des informations sur la sécurité au niveau des lignes, consultez Sécurité au niveau des lignes.

enable_numeric_rounding

Valeurs (par défaut en gras)

on (true), off (false)

Description

Spécifie s'il faut utiliser l'arrondissement numérique. Si `enable_numeric_rounding` est on, Amazon Redshift arrondit les valeurs NUMERIC lorsqu'il les convertit en d'autres types numériques, tels que INTEGER ou DECIMAL. Si `enable_numeric_rounding` est `off`, Amazon Redshift tronque les valeurs NUMERIC lorsqu'il les convertit en d'autres types numériques. Pour plus d'informations sur les types numériques, consultez Types numériques.

exemple

```
--Create a table and insert the numeric value 1.5 into it.
CREATE TABLE t (a numeric(10, 2));

INSERT INTO t VALUES (1.5);

SET enable_numeric_rounding to ON;
--Amazon Redshift now rounds NUMERIC values when casting to other numeric types.

SELECT a::int FROM t;

 a
---
 2
(1 row)


SELECT a::decimal(10, 0) FROM t;

 a
---
 2
(1 row)


 SELECT a::decimal(10, 5) FROM t;

    a
---------
 1.50000
(1 row)


SET enable_numeric_rounding to OFF;
--Amazon Redshift now truncates NUMERIC values when casting to other numeric types.

SELECT a::int FROM t;

 a
---
 1
(1 row)
```

```
SELECT a::decimal(10, 0) FROM t;

 a
---
 1
(1 row)


SELECT a::decimal(10, 5) FROM t;

    a
---------
 1.50000
(1 row)
```

enable_result_cache_for_session

Valeurs (par défaut en gras)

on (vrai), off (faux)

Description

Spécifie si la mise en cache des résultats des requêtes doit être utilisée. Si `enable_result_cache_for_session` est on, Amazon Redshift recherche une copie valide mise en cache des résultats de la requête lorsqu'une requête est soumise. Si une correspondance est trouvée dans le cache des résultats, Amazon Redshift utilise les résultats mis en cache et n'exécute pas la requête. Si `enable_result_cache_for_session` est `off`, Amazon Redshift ignore le cache des résultats et exécute toutes les requêtes lorsqu'elles sont soumises.

exemple

```
SET enable_result_cache_for_session TO off;
--Amazon Redshift now ignores the results cache
```

enable_vacuum_boost

Valeurs (par défaut en gras)

false, true

Description

Spécifie si l'option vacuum boost doit être activée pour toutes les commandes VACUUM exécutées pendant une séance. Si `enable_vacuum_boost` est `true`, Amazon Redshift exécute toutes les commandes VACUUM de la séance avec l'option BOOST. Si `enable_vacuum_boost` est `false`, Amazon Redshift n'exécute pas l'option BOOST par défaut. Pour plus d'informations sur l'option BOOST, consultez VACUUM.

error_on_nondeterministic_update

Valeurs (par défaut en gras)

false, true

Description

Spécifie si les requêtes UPDATE avec plusieurs correspondances par ligne génèrent une erreur.

exemple

```
SET error_on_nondeterministic_update TO true;

CREATE TABLE t1(x1 int, y1 int);

CREATE TABLE t2(x2 int, y2 int);

INSERT INTO t1 VALUES (1,10), (2,20), (3,30);

INSERT INTO t2 VALUES (2,40), (2,50);

UPDATE t1 SET y1=y2 FROM t2 WHERE x1=x2;

ERROR: Found multiple matches to update the same tuple.
```

extra_float_digits

Valeurs (par défaut en gras)

0, -15 à 2

Description

Définit le nombre de chiffres affichés pour les valeurs à virgule flottante, y compris float4 et float8. La valeur est ajoutée au nombre standard de chiffres (FLT_DIG ou DBL_DIG le cas échéant). La valeur peut être définie sur 2 (valeur maximale) pour inclure des chiffres partiellement significatifs. Cela est particulièrement utile quand il s'agit de générer des données flottantes qui doivent être restaurées à l'identique. Ou elle peut être négative pour supprimer les chiffres indésirables.

exemple

L'exemple suivant définit `extra_float_digits` sur `-2`. Tout d'abord, affichez le réglage actuel des paramètres.

```
show all;
      name                      | setting
--------------------------+----------------
analyze_threshold_percent | 10
datestyle                 | ISO, MDY
extra_float_digits        | 2
query_group               | default
search_path               | $user, public
statement_timeout         | 0
timezone                  | UTC
wlm_query_slot_count      | 1
```

Définissez ensuite la nouvelle valeur sur `-2`.

```
set extra_float_digits to -2;
```

Enfin, affichez le réglage des paramètres mis à jour.

```
show all;
      name                      | setting
--------------------------+----------------
```

```
analyze_threshold_percent | 10
datestyle                 | ISO, MDY
extra_float_digits        | -2
query_group               | default
search_path               | $user, public
statement_timeout         | 0
timezone                  | UTC
wlm_query_slot_count      | 1
```

interval_forbid_composite_literals

Valeurs (par défaut en gras)

faux, vrai

Description

Configuration de session qui modifie la valeur d'un intervalle contenant à la fois les parties ANNÉE PAR MOIS et JOUR PAR SECONDE.

Si `interval_forbid_composite_literals` tel est le cas`true`, une erreur est renvoyée si un intervalle comportant à la fois les parties ANNÉE PAR MOIS et JOUR PAR SECONDE est rencontré. Par exemple, le code SQL suivant contient un INTERVAL DAY TO SECOND avec des parties YEAR TO MONTH et DAY TO SECOND.

```
SELECT INTERVAL '1 year 1 day' DAY TO SECOND;
ERROR:  Interval Day To Second literal cannot contain year-month parts. Disable the GUC
 interval_forbid_composite_literals to suppress this error and silently discard the
 year-month part.
```

Si tel `interval_forbid_composite_literals` est `false` le cas, Amazon Redshift supprime une erreur et tronque la partie YEAR TO MONTH d'une valeur INTERVAL DAY TO SECOND. Par exemple, le code SQL suivant contient un INTERVAL DAY TO SECOND avec des parties YEAR TO MONTH et DAY TO SECOND.

```
SET interval_forbid_composite_literals to "false";
SELECT INTERVAL '1 year 1 day' DAY TO SECOND;


 intervald2s
```

```
-----------------------------
1 days 0 hours 0 mins 0.0 secs
```

json_serialization_enable

Valeurs (par défaut en gras)

false, true

Description

Configuration de séance qui modifie le comportement de sérialisation JSON des données formatées ORC, JSON, Ion et Parquet. Si `json_serialization_enable` est `true`, toutes les collections de niveau supérieur sont automatiquement sérialisées en JSON et renvoyées en tant que VARCHAR (65535). Les colonnes non complexes ne sont pas affectées ou sérialisées. Les colonnes de collection étant sérialisées au format VARCHAR (65535), il n'est plus possible d'accéder directement à leurs sous-champs imbriqués dans la syntaxe de la requête (c'est-à-dire dans la clause de filtrage). Si `json_serialization_enable` est `false`, les collections de niveau supérieur ne sont pas sérialisées en JSON. Pour plus d'informations sur la sérialisation JSON imbriquée, consultez Sérialisation de JSON imbriqué complexe.

json_serialization_parse_nested_strings

Valeurs (par défaut en gras)

false, true

Description

Configuration de séance qui modifie le comportement de sérialisation JSON des données formatées ORC, JSON, Ion et Parquet. Lorsque les conditions `json_serialization_parse_nested_strings` et `json_serialization_enable` sont remplies, les valeurs de chaîne stockées dans des types complexes (tels que les cartes, les structures ou les tableaux) sont analysées et écrites directement en ligne dans le résultat s'il s'agit de JSON valide. Si `json_serialization_parse_nested_strings` est false, les chaînes dans les types complexes imbriqués sont sérialisées en tant que chaînes JSON échappées. Pour plus d'informations, consultez Sérialisation de types complexes contenant des chaînes JSON.

max_concurrency_scaling_clusters

Valeurs (par défaut en gras)

1, 0 à 10

Description

Définit le nombre maximum de clusters d'adaptation de la simultanéité autorisé lorsque la mise à l'échelle de la simultanéité est activée. Augmentez cette valeur si davantage de mise à l'échelle de la simultanéité est requise. Diminuez cette valeur pour réduire l'utilisation des clusters de mise à l'échelle de la simultanéité et la facturation qui en résulte.

Le nombre maximal de clusters de mise à l'échelle de la simultanéité est un quota réglable. Pour plus d'informations, consultez la rubrique Quotas Amazon Redshift dans le Guide de gestion Amazon Redshift.

max_cursor_result_set_size

Valeurs (par défaut en gras)

0 (valeur maximale par défaut) - 14400000 Mo

Description

Le paramètre `max_cursor_result_set_size` n'est plus utilisé. Pour plus d'informations sur cursor_result_set_size, consultez Contraintes de curseur.

mv_enable_aqmv_for_session

Valeurs (par défaut en gras)

true, false

Description

Spécifie si Amazon Redshift peut effectuer une réécriture automatique des requêtes des vues matérialisées au niveau de la séance.

navigate_super_null_on_error

Valeurs (par défaut en gras)

on, off

Description

Spécifie que lorsque vous essayez de naviguer vers un membre inexistant d'un objet ou d'un élément d'un tableau, Amazon Redshift renvoie une valeur NULL si votre requête est exécutée en mode par défaut « lax ».

parse_super_null_on_error

Valeurs (par défaut en gras)

off, on

Description

Spécifie que lorsqu'Amazon Redshift essaie d'analyser un membre inexistant d'un objet ou d'un élément d'un tableau, Amazon Redshift renvoie une valeur NULL si votre requête est exécutée en mode « strict ».

pg_federation_repeatable_read

Valeurs (par défaut en gras)

true, false

Description

Spécifie le niveau d'isolation des transactions de la requête fédérée pour les résultats de la base de données PostgreSQL.

- Quand `pg_federation_repeatable_read` a pour valeur true, les transactions fédérées sont traitées avec la sémantique du niveau d'isolation REPEATABLE READ. Il s’agit de l’option par défaut.

- Lorsque le paramètre pg_federation_repeatable_read est défini comme faux, les transactions fédérées sont traitées avec la sémantique du niveau d'isolation READ COMMITTED.

Pour plus d'informations, consultez les ressources suivantes :

- Éléments à prendre en compte lors de l'accès aux données fédérées avec Amazon Redshift.
- Gestion des opérations d'écriture simultanées.

Exemples

La commande suivante permet de définir pg_federation_repeatable_read sur on pour une session. La commande show montre la valeur de la valeur définie.

```
set pg_federation_repeatable_read to on;
```

```
show pg_federation_repeatable_read;

pg_federation_repeatable_read
-----------------------------
on
```

query_group

Valeurs (par défaut en gras)

Aucune valeur par défaut ; la valeur peut être n'importe quelle chaîne de caractères.

Description

Applique une étiquette définie par l'utilisateur à un groupe de requêtes exécutées lors de la même séance. Cette étiquette est capturée dans les journaux de requêtes. Vous pouvez l'utiliser pour limiter les résultats issus des tables STL_QUERY et STV_INFLIGHT et de la vue SVL_QLOG. Par exemple, vous pouvez appliquer une étiquette distincte à chaque requête que vous exécutez afin d'identifier les requêtes de manière unique sans avoir à les rechercher IDs.

Ce paramètre n'existe pas dans le fichier de configuration de serveur et doit être défini lors de l'exécution avec une commande SET. Même si vous pouvez utiliser une longue chaîne de caractères

comme étiquette, l'étiquette est tronquée à 30 caractères dans la colonne LABEL de la table STL_QUERY et de la vue SVL_QLOG (et à 15 caractères dans STV_INFLIGHT).

Dans l'exemple suivant, query_group est défini sur **`Monday`**, puis plusieurs requêtes sont exécutées avec cette étiquette.

```
set query_group to 'Monday';
SET
select * from category limit 1;
...
...
select query, pid, substring, elapsed, label
from svl_qlog where label ='Monday'
order by query;

query | pid  |        substring                   |  elapsed  | label
------+------+------------------------------------+-----------+--------
789   | 6084 | select * from category limit 1;    | 65468     | Monday
790   | 6084 | select query, trim(label) from ... | 1260327   | Monday
791   | 6084 | select * from svl_qlog where ..    | 2293547   | Monday
792   | 6084 | select count(*) from bigsales;     | 108235617 | Monday
...
```

search_path

Valeurs (par défaut en gras)

'$user', public, schema_names

Liste séparée par des virgules de noms de schéma existants. Si '$user' est présent, le schéma ayant le même nom que `SESSION_USER` est remplacé, sinon il est ignoré.

Description

Spécifie l'ordre dans lequel les schémas sont explorés lorsqu'un objet (une table ou une fonction, par exemple) est référencé par un nom simple sans composant de schéma :

- Les chemins de recherche ne sont pas pris en charge dans les schémas et tables externes. Les tables externes doivent être explicitement qualifiées à l'aide d'un schéma externe. Les partages de données se comportent comme des données externes et ne sont donc pas pris en charge en tant que chemins de recherche.

- Lorsque les objets sont créés sans schéma cible spécifique, ils sont placés dans le premier schéma répertorié dans le chemin de recherche. Si le chemin de recherche est vide, le système renvoie une erreur.
- Lorsque des objets avec des noms identiques existent dans des schémas différents, le premier trouvé dans le chemin de recherche est utilisé.
- Un objet qui n'est pas dans l'un des schémas du chemin de recherche peut uniquement être référencé en spécifiant son schéma contenant avec un nom complet (en pointillé).
- Le schéma du catalogue système, pg_catalog, est toujours exploré. S'il est mentionné dans le chemin d'accès, il est exploré dans l'ordre spécifié. Dans le cas contraire, il est exploré avant l'un des éléments du chemin d'accès.
- Le schéma des tables temporaires de la séance en cours, pg_temp_nnn, est toujours exploré s'il existe. Il peut être explicitement répertorié dans le chemin d'accès à l'aide de l'alias pg_temp. S'il n'apparaît pas dans le chemin d'accès, il est exploré en premier (avant même pg_catalog). Cependant, le schéma temporaire n'est exploré que pour les noms de relation (tables, vues). Il n'est pas exploré pour les noms de fonction.

exemple

L'exemple suivant crée le schéma ENTERPRISE et définit le search_path vers le nouveau schéma.

```
create schema enterprise;
set search_path to enterprise;
show search_path;

 search_path
-------------
 enterprise
(1 row)
```

L'exemple suivant ajoute le schéma ENTERPRISE au search_path par défaut.

```
set search_path to '$user', public, enterprise;
show search_path;

         search_path
-----------------------------
 "$user", public, enterprise
(1 row)
```

L'exemple suivant ajoute la table FRONTIER au schéma ENTERPRISE.

```
create table enterprise.frontier (c1 int);
```

Lorsque la table PUBLIC.FRONTIER est créée dans la même base de données et que l'utilisateur ne spécifie pas le nom du schéma dans une requête, PUBLIC.FRONTIER a priorité sur ENTERPRISE.FRONTIER.

```
create table public.frontier(c1 int);
insert into enterprise.frontier values(1);
select * from frontier;

frontier
----
(0 rows)

select * from enterprise.frontier;

c1
----
1
(1 row)
```

spectrum_enable_pseudo_columns

Valeurs (par défaut en gras)

true, false

Description

Vous pouvez désactiver la création de pseudo-colonnes d'une séance en définissant le paramètre de configuration `spectrum_enable_pseudo_columns` avec la valeur `false`.

exemple

La commande suivante désactive la création de pseudo-colonnes d'une séance.

```
set spectrum_enable_pseudo_columns to false;
```

enable_spectrum_oid

Valeurs (par défaut en gras)

true, false

Description

Vous pouvez également désactiver uniquement la pseudo-colonne `$spectrum_oid` en définissant le paramètre de configuration `enable_spectrum_oid` avec la valeur `false`.

exemple

La commande suivante désactive la pseudo-colonne `$spectrum_oid` en définissant le paramètre de configuration `enable_spectrum_oid` avec la valeur `false`.

```
set enable_spectrum_oid to false;
```

spectrum_query_maxerror

Valeurs (par défaut en gras)

-1, entier

Description

Vous pouvez spécifier un entier pour indiquer le nombre maximal d'erreurs à accepter avant d'annuler la requête. Une valeur négative désactive le traitement maximal des données d'erreur. Les résultats sont enregistrés dans SVL_SPECTRUM_SCAN_ERROR.

exemple

L'exemple suivant suppose que les données ORC contiennent des caractères excédentaires et des caractères non valides. La définition de la colonne pour `my_string` spécifie une longueur de 3 caractères. Voici un exemple de données pour cet exemple :

```
my_string
```

```
---------------
abcdef
gh
ab
```

Les commandes suivantes définissent le nombre maximal d'erreurs sur 1 et exécutent la requête.

```
set spectrum_query_maxerror to 1;
SELECT my_string FROM orc_data;
```

La requête s'arrête et les résultats sont enregistrés dans SVL_SPECTRUM_SCAN_ERROR.

statement_timeout

Valeurs (par défaut en gras)

0 (désactive la limitation), x millisecondes

Description

Interrompt toute instruction qui dépasse le nombre de millisecondes spécifié.

La valeur de `statement_timeout` est la durée maximale d'exécution d'une requête avant qu'Amazon Redshift n'y mette fin. Cette durée inclut la planification, les files d'attente WLM et le délai d'exécution. Comparez cette durée au délai WLM (max_execution_time) et au QMR (query_execution_time), qui incluent uniquement le délai d'exécution.

Si le délai WLM (max_execution_time) est également spécifié dans le cadre d'une configuration WLM, la valeur la plus basse de statement_timeout et de max_execution_time est utilisée. Pour plus d'informations, consultez Délai WLM.

exemple

Comme la requête suivante dure plus d'1 milliseconde, elle expire et est annulée.

```
set statement_timeout = 1;

select * from listing where listid>5000;
```

```
ERROR:  Query (150) canceled on user's request
```

stored_proc_log_min_messages

Valeurs (par défaut en gras)

LOG, INFO, AVIS, AVERTISSEMENT, EXCEPTION

Description

Spécifie le niveau minimal de journalisation des messages de procédure stockée soulevés. Les messages au niveau ou au-dessus du niveau spécifié sont enregistrés. La valeur par défaut est LOG (tous les messages sont enregistrés). L'ordre des niveaux de journalisation du plus haut au plus bas est :

1. EXCEPTION
2. WARNING
3. NOTICE
4. INFO
5. LOG

Par exemple, si vous spécifiez une valeur AVIS, les messages sont uniquement consignés pour AVIS, AVERTISSEMENT et EXCEPTION.

timezone

Valeurs (par défaut en gras)

UTC, fuseau horaire

Syntaxe

```
SET timezone { TO | = } [ time_zone | DEFAULT ]

SET time zone [ time_zone | DEFAULT ]
```

Description

Définit le fuseau horaire de la séance en cours. Le fuseau horaire peut correspondre au décalage par rapport à l'heure UTC (Coordinated Universal Time) ou un nom de fuseau horaire.

> **Note**
>
> Vous ne pouvez pas définir le paramètre de configuration `timezone` à l'aide d'un groupe de paramètres de cluster. Vous pouvez définir le fuseau horaire uniquement pour la séance actuelle en utilisant une commande SET. Pour définir le fuseau horaire pour toutes les séances exécutées par un utilisateur de base de données spécifique, utilisez la commande ALTER USER. ALTER USER … SET TIMEZONE modifie le fuseau horaire pour les séances ultérieures, et non pour la séance en cours.

Lorsque vous définissez le fuseau horaire à l'aide de la commande `SET timezone` (un mot) avec TO ou =, vous pouvez spécifier time_zone comme nom de fuseau horaire, décalage de format de style POSIX ou décalage de format ISO-8601, comme illustré ci-après.

```
SET timezone { TO | = } time_zone
```

Lorsque vous définissez le fuseau horaire à l'aide de la commande SET time zone sans TO ou =, vous pouvez également spécifier time_zone à l'aide d'un INTERVALLE comme nom de fuseau horaire, décalage de format de style POSIX ou décalage de format ISO-8601, comme illustré ci-après.

```
SET time zone time_zone
```

Formats de fuseau horaire

Amazon Redshift prend en charge les formats de fuseau horaire suivants :

- Nom de fuseau horaire
- INTERVAL
- Spécification de fuseau horaire POSIX
- Décalage ISO-8601

Comme les abréviations de fuseaux horaires, telles que PST ou PDT, sont définies comme un décalage fixe de l'UTC et n'incluent pas de règles de l'heure d'été, la commande SET ne prend pas en charge les abréviations de fuseaux horaires.

Pour plus d'informations sur les formats de fuseau horaire, consultez les éléments suivants.

Nom du fuseau horaire – Nom complet du fuseau horaire, tel qu'Amérique/New_York. Les noms de fuseaux horaires complets peuvent comprendre des règles de l'heure d'été.

Voici des exemples de noms de fuseaux horaires :

- Etc/Greenwich
- Amérique/New_York
- CST6CDT
- Go

> **Note**
>
> Plusieurs noms de fuseaux horaires, comme EST, MST, NZ et UCT, sont également des abréviations.

Pour afficher la liste des noms de fuseaux horaires valides pris en charge, exécutez la commande suivante.

```
select pg_timezone_names();
```

INTERVALLE – Décalage de l'UTC. Par exemple, PST est –8:00 ou –8 heures.

Voici des exemples de décalages de fuseau horaire avec INTERVALLE :

- –8:00
- –8 heures
- 30 minutes

Format de style POSIX — Spécification de fuseau horaire au format STDoffsetSTDoffsetDST, où STD est une abréviation de fuseau horaire, offset est le décalage numérique en heures à l'ouest par

rapport à l'UTC et DST est une abréviation facultative de zone d'été. L'heure d'été est supposée être une heure d'avance par rapport au décalage donné.

Les formats de fuseaux horaires de style POSIX utilisent des décalages positifs à l'ouest de Greenwich, contrairement à la convention ISO-8601, qui utilise des décalages positifs à l'est de Greenwich.

Voici des exemples de fuseaux horaires de style POSIX :

- PST8
- PST8PDT
- EST5
- EST5EDT

> **Note**
>
> Amazon Redshift ne valide pas les spécifications de fuseaux horaires de style POSIX, il est donc possible définir le fuseau horaire sur une valeur non valide. Par exemple, la commande suivante ne renvoie pas d'erreur, même si elle définit le fuseau horaire sur une valeur non valide.
>
> ```
> set timezone to 'xxx36';
> ```

Décalage ISO-8601 – Décalage d'UTC sous la forme `±[hh]:[mm]`.

Voici quelques exemples de décalages ISO-8601 :

- -8:00
- +7:30

Exemples

L'exemple suivant définit le fuseau horaire de la séance en cours sur New York.

```
set timezone = 'America/New_York';
```

L'exemple suivant définit le fuseau horaire de la séance en cours sur UTC–8 (PST).

```
set timezone to '-8:00';
```

L'exemple suivant utilise l'INTERVALLE pour définir le fuseau horaire sur PST.

```
set timezone interval '-8 hours'
```

L'exemple suivant réinitialise le fuseau horaire de la séance en cours sur le fuseau horaire système par défaut (UTC).

```
set timezone to default;
```

Pour définir le fuseau horaire de l'utilisateur de la base de données, utilisez l'instruction ALTER USER … SET. L'exemple suivant définit le fuseau horaire de l'utilisateur dbuser sur New York. Cette nouvelle valeur persiste pour toutes les séances ultérieures de cet utilisateur.

```
ALTER USER dbuser SET timezone to 'America/New_York';
```

use_fips_ssl

Valeurs (par défaut en gras)

true, false

Description

Une valeur de groupe de paramètres qui indique si le mode SSL conforme à la norme FIPS est utilisé. Si tel `use_fips_ssl` est le cas`true`, le mode SSL conforme à la norme FIPS est utilisé. Si tel `use_fips_ssl` est le cas`false`, le mode SSL conforme à la norme FIPS n'est pas utilisé. Pour plus d'informations, consultez la section Configuration des options de sécurité pour les connexions dans le guide de gestion Amazon Redshift.

Pour configurer les paramètres d'un cluster provisionné par Amazon Redshift, consultez la section À propos des groupes de paramètres dans le guide de gestion Amazon Redshift. Pour configurer les paramètres de Redshift Serverless, consultez la section Configuration d'une connexion SSL

compatible FIPS à Amazon Redshift Serverless dans le guide de gestion Amazon Redshift et/ou dans le manuel de référence des API Redshift Serverless. CreateWorkgroupUpdateWorkgroup

wlm_query_slot_count

Valeurs (par défaut en gras)

1, 1 à 50 (ne peut pas dépasser le nombre d'emplacements disponibles (niveau de concurrence) pour la classe de service)

Description

Définit le nombre d'emplacements de requête qu'une requête utilise.

La gestion de l'application (WLM) réserve des emplacements dans une classe de service selon le niveau de simultanéité défini pour la file d'attente. Par exemple, si le niveau de concurrence est défini sur 5, la classe de service dispose de 5 emplacements. WLM alloue la mémoire disponible pour une classe de service de façon égale pour chaque emplacement. Pour plus d'informations, consultez Gestion de la charge de travail.

> **Note**
>
> Si la valeur de wlm_query_slot_count est supérieure au nombre d'emplacements disponibles (niveau de concurrence) pour la classe de service, la requête échoue. Si vous rencontrez une erreur, diminuez wlm_query_slot_count à une valeur admissible.

Pour les opérations où les performances sont fortement affectées par la quantité de mémoire allouée, telles que l'opération VACUUM, l'augmentation de la valeur de wlm_query_slot_count peut améliorer les performances. En particulier, pour les commandes VACUUM lentes, examinez l'enregistrement correspondant dans la vue SVV_VACUUM_SUMMARY. Si vous voyez des valeurs élevées (proches ou supérieures à 100) pour sort_partitions et merge_increments de la vue SVV_VACUUM_SUMMARY, envisagez d'augmenter la valeur de wlm_query_slot_count la prochaine fois où vous exécuterez une opération VACUUM sur cette table.

L'augmentation de la valeur de wlm_query_slot_count limite le nombre de requêtes simultanées qui peuvent être exécutées. Par exemple, supposons que la classe de service possède un niveau de concurrence de 5 et que wlm_query_slot_count ait la valeur 3. Lorsqu'une requête est en cours

d'exécution dans la séance avec wlm_query_slot_count défini sur 3, un maximum de 2 requêtes concurrentes supplémentaires peuvent être exécutées dans la même classe de service. Les requêtes suivantes restent dans la file d'attente tant que les requêtes en cours d'exécution n'ont pas abouti et que les emplacements n'ont pas été libérés.

Exemples

Utilisez la commande SET pour définir la valeur de wlm_query_slot_count pour la durée de la séance en cours.

```
set wlm_query_slot_count to 3;
```

Historique du document

> **Note**
>
> Pour obtenir une description des nouvelles fonctionnalités d'Amazon Redshift, consultez la section Nouveautés.

Le tableau suivant décrit les modifications importantes apportées à la documentation du guide du développeur de base de données Amazon Redshift après mai 2018. Pour recevoir les notifications de mise à jour de cette documentation, abonnez-vous à un flux RSS.

Version de l'API : 2012-12-01

Pour obtenir la liste des modifications apportées au Guide de gestion Amazon Redshift, veuillez consulter l'historique du document.

Pour plus d'informations sur les nouvelles fonctions, y compris une liste des correctifs et les numéros de version de cluster associés pour chaque version de produit, consultez le Historique des versions de cluster.

Modification	Description	Date
Prise en charge de géométries spatiales 3D et 4D et des nouvelles fonctions spatiales	Vous pouvez désormais utiliser des fonctions spatiales supplémentaires et la prise en charge de la géométrie 3D et 4D est ajoutée à certaines fonctions.	19 août 2021
Prise en charge de l'encodage de compression d'une colonne pour l'optimisation automatique des tables	Vous pouvez spécifier l'option ENCODE AUTO pour une table afin de permettre à Amazon Redshift de gérer automatiquement l'encodage de la compression pour toutes les colonnes de la table.	3 août 2021

Prise en charge de plusieurs instructions SQL ou d'une instruction SQL avec des paramètres à l'aide de l'API de données Amazon Redshift	Vous pouvez à présent exécuter plusieurs instructions SQL ou une instruction avec des paramètres à l'aide de l'API de données Amazon Redshift.	28 juillet 2021
Prise en charge du classement insensible à la casse avec des remplacements au niveau des colonnes	Vous pouvez désormais utiliser la clause COLLATE dans une instruction CREATE DATABASE pour spécifier le classement par défaut.	24 juin 2021
Prise en charge du partage de données entre comptes	Vous pouvez désormais partager des données entre les deux Comptes AWS.	30 avril 2021
Prise en charge des requêtes de données hiérarchiques avec CTE récursif	Vous pouvez désormais utiliser une expression de table commune récursive (CTE) dans votre SQL.	29 avril 2021
Prise en charge des requêtes entre bases de données	Vous pouvez désormais interroger des données entre les bases de données d'un cluster.	10 mars 2021
Prise en charge d'un contrôle précis des accès sur les commandes COPY et UNLOAD	Vous pouvez désormais accorder le privilège d'exécution des commandes COPY et UNLOAD à des utilisateurs et groupes spécifiques de votre cluster Amazon Redshift afin de créer une stratégie de contrôle d'accès plus précise.	12 janvier 2021

Prise en charge du JSON natif et des données semi-structurées	Vous pouvez désormais définir le type de données SUPER.	9 décembre 2020
Prise en charge des requêtes fédérées vers MySQL	Vous pouvez désormais écrire une requête fédérée sur un moteur MySQL pris en charge.	9 décembre 2020
Prise en charge du partage de données	Vous pouvez désormais partager des données entre des clusters Amazon Redshift.	9 décembre 2020
Prise en charge de l'optimisation automatique des tables	Vous pouvez désormais définir des clés de distribution et de tri automatiques.	9 décembre 2020
Prise en charge d'Amazon Redshift ML	Vous pouvez désormais créer, entraîner et déployer des modèles de machine learning (ML).	8 décembre 2020
Prise en charge de l'actualisation automatique et de la réécriture des requêtes des vues matérialisées	Vous pouvez désormais conserver les vues matérialisées up-to-date grâce à l'actualisation automatique et les performances des requêtes peuvent être améliorées grâce à la réécriture automatique.	11 novembre 2020
Prise en charge des types de données TIME et TIMETZ	Vous pouvez désormais créer des tables avec des types de données TIME et TIMETZ. Le type de données TIME stocke l'heure du jour sans fuseau horaire tandis que TIMETZ stocke l'heure du jour, dont le fuseau horaire	11 novembre 2020

Support pour Lambda UDFs et tokenisation	Vous pouvez désormais écrire Lambda UDFs pour activer la tokenisation externe des données.	26 octobre 2020
Prise en charge de la modification du codage d'une colonne de table	Vous pouvez désormais modifier le codage d'une colonne de table.	20 octobre 2020
Prise en charge des requêtes entre les bases de données	Amazon Redshift peut désormais interroger plusieurs bases de données d'un cluster.	15 octobre 2020
Support pour les HyperLogLog croquis	Amazon Redshift peut désormais stocker et traiter. HyperLogLogSketches	2 octobre 2020
Prise en charge d'Apache Hudi et de Delta Lake	Des améliorations ont été apportées à la création de tables externes pour Redshift Spectrum.	24 septembre 2020
Prise en charge des améliorations apportées à l'interrogation des données spatiales	Les améliorations incluent le chargement d'un fichier de formes et plusieurs nouvelles fonctions SQL spatiales.	15 septembre 2020
Les vues matérialisées prennent en charge les tables externes	Vous pouvez créer des vues matérialisées dans Amazon Redshift qui font référence à des sources de données externes.	19 juin 2020

Prise en charge de l'écriture dans une table externe	Vous pouvez écrire dans des tables externes en exécutant la commande CREATE EXTERNAL TABLE AS SELECT pour écrire dans une nouvelle table externe ou la commande INSERT INTO pour insérer des données dans une table externe existante.	8 juin 2020
Prise en charge des contrôles de stockage pour les schémas	Les commandes et vues qui gèrent les contrôles de stockage pour les schémas ont été mises à jour.	2 juin 2020
Prise en charge de la disponibilité générale des requêtes fédérées	Mise à jour des informations sur l'interrogation de données avec des requêtes fédérées.	16 avril 2020
Prise en charge de fonctions spatiales supplémentaires	Ajout de descriptions de fonctions spatiales supplémentaires.	2 avril 2020
Disponibilité générale de la prise en charge des vues matérialisées	La disponibilité générale des vues matérialisées est effective à partir de la version de cluster 1.0.13059.	19 février 2020
Prise en charge des privilèges de niveau colonne	Les privilèges de niveau colonne sont disponibles à partir de la version de cluster 1.0.13059.	19 février 2020

ALTER TABLE	Vous pouvez utiliser une commande ALTER TABLE avec la clause ALTER DISTSTYLE ALL pour modifier le style de distribution d'une table.	11 février 2020
Prise en charge des requêtes fédérées	Mise à jour du guide pour décrire des requêtes fédérées avec une commande CREATE EXTERNAL SCHEMA mise à jour.	3 décembre 2019
Prise en charge de l'exportation de lac de données	Mise à jour du guide pour décrire les nouveaux paramètres de la commande UNLOAD.	3 décembre 2019
Prise en charge des données spatiales	Manuel mis à jour pour décrire la prise en charge des données spatiales	21 novembre 2019
Prise en charge de la nouvelle console	Manuel mis à jour pour décrire la nouvelle console Amazon Redshift.	11 novembre 2019
Prise en charge du tri automatique des tables	Amazon Redshift peut trier automatiquement les données des tables.	7 novembre 2019
Prise en charge de l'option VACUUM BOOST	Vous pouvez utiliser l'option BOOST lorsque vous nettoyez les tables.	7 novembre 2019
Prise en charge des colonnes IDENTITY par défaut	Vous pouvez créer des tables avec des colonnes IDENTITY par défaut.	19 septembre 2019

Support pour le codage AZ64 par compression	Vous pouvez encoder certaines colonnes par AZ64 compression.	19 septembre 2019
Prise en charge de la priorité de requête	Vous pouvez définir la priorité de requête d'une file d'attente WLM automatique.	22 août 2019
Support pour AWS Lake Formation	Vous pouvez utiliser un catalogue de données Lake Formation avec Amazon Redshift Spectrum.	8 août 2019
COMPUPDATE PRESET	Vous pouvez utiliser une commande COPY avec COMPUPDATE PRESET pour permettre à Amazon Redshift de choisir l'encodage de compression.	13 juin 2019
ALTER COLUMN	Vous pouvez utiliser une commande ALTER TABLE avec ALTER COLUMN pour augmenter la taille d'une colonne VARCHAR.	22 mai 2019
Prise en charge des procédures stockées	Vous pouvez définir les procédures stockées PL/pgSQL dans Amazon Redshift.	24 avril 2019
Prise en charge de la configuration de la gestion automatique de la charge de travail (WLM)	Vous pouvez permettre à Amazon Redshift de fonctionner en WLM automatique.	24 avril 2019

Commande UNLOAD pour compression Zstandard	Vous pouvez utiliser la commande UNLOAD pour appliquer la compression Zstandard aux fichiers texte et aux fichiers CSV (comma-se parated value) déchargés sur Amazon S3.	3 avril 2019
Mise à l'échelle de la simultanéité	Lorsque la mise à l'échelle de la simultanéité est activée, Amazon Redshift ajoute automatiquement de la capacité de cluster supplémen taire lorsque vous en avez besoin pour traiter une augmentation des requêtes de lecture simultanées.	21 mars 2019
UNLOAD vers CSV	Vous pouvez utiliser la commande UNLOAD pour décharger vers un fichier au format CSV.	13 mars 2019
Style de distribution AUTO	Pour activer la distribution automatique, vous pouvez spécifier le style de distribut ion AUTO avec un énoncé CREATE TABLE. Lorsque vous activez la distribution automatique, Amazon Redshift attribue un style de distribut ion optimal en fonction des données de la table. Le changement de distribution se produit en arrière-plan, en quelques secondes.	23 janvier 2019

La commande COPY depuis Parquet prend en charge SMALLINT	La commande COPY prend désormais en charge le chargement de fichiers au format Parquet en colonnes qui utilisent le type de données SMALLINT. Pour plus d'informations, consultez COPY à partir de Formats de données en colonnes.	2 janvier 2019
DROP EXTERNAL DATABASE	Vous pouvez supprimer une base de données externe en incluant la clause DROP EXTERNAL DATABASE avec une commande DROP SCHEMA.	3 décembre 2018
Opération UNLOAD entre régions	Vous pouvez DÉCHARGER dans un compartiment Amazon S3 d'une autre AWS région en spécifiant le paramètre REGION.	31 octobre 2018
Suppression de vide automatique	Amazon Redshift exécute automatiquement une opération VACUUM DELETE en arrière-plan, de sorte que vous avez rarement, voire jamais, besoin d'exécuter un DELETE ONLY vacuum. Amazon Redshift planifie l'exécution de VACUUM DELETE pendant les périodes de charge réduite et interrompt l'opération pendant les périodes de charge élevée.	31 octobre 2018

Distribution automatique	Lorsque vous ne spécifiez pas de style de distribution avec une instruction CREATE TABLE, Amazon Redshift attribue un style de distribution optimal basé sur les données de la table. Le changement de distribution se produit en arrière-plan, en quelques secondes.	31 octobre 2018
Contrôle d'accès détaillé pour AWS Glue Data Catalog	Vous pouvez désormais définir les niveaux d'accès aux données stockées dans le AWS Glue Data Catalog.	15 octobre 2018
Opération UNLOAD avec types de données	Vous pouvez spécifier l'option MANIFEST VERBOSE avec une commande UNLOAD pour ajouter les métadonnées au fichier manifeste, y compris les noms et les types de données des colonnes, les tailles de fichiers et les nombres de lignes.	10 octobre 2018
Ajout de plusieurs partitions à l'aide d'une seule instruction ALTER TABLE	Pour les tables externes Redshift Spectrum, vous pouvez combiner plusieurs clauses PARTITION en une seule instruction ALTER TABLE ADD. Pour plus d'informations, consultez Exemples de modification d'une table externe.	10 octobre 2018

UNLOAD avec en-tête	Vous pouvez spécifier l'option HEADER avec une commande UNLOAD pour ajouter une ligne d'en-tête contenant des noms de colonne en haut de chaque fichier de sortie.	19 septembre 2018
Nouvelles vues et table système	De la documentation sur SVL_S3Retries, SVL_USER_INFO et STL_DISK_FULL_DIAG a été ajoutée.	31 août 2018
Prise en charge de données imbriquées dans Amazon Redshift Spectrum	Vous pouvez désormais soumettre une requête de données imbriquées dans les tables Amazon Redshift Spectrum. Pour plus d'informations, consultez Didacticiel : Faire une requête de données imbriquées avec Amazon Redshift Spectrum.	8 août 2018
ARC activé par défaut.	L'accélération de requête courte (ARC) est désormais activée par défaut pour tous les nouveaux clusters. L'ARC utilise le machine learning afin d'offrir des performances supérieures, des résultats plus rapides et une meilleure prévisibilité des temps d'exécution des requêtes. Pour plus d'informations, voir Accélération de requête courte.	8 août 2018

Amazon Redshift Advisor	Vous pouvez désormais obtenir des recommandations personnalisées sur la façon d'améliorer les performances de votre cluster et de réduire les coûts de fonctionnement à partir d'Amazon Redshift Advisor. Pour plus d'informations, consultez Amazon Redshift Advisor.	26 juillet 2018
Référence d'alias immédiate	Vous pouvez désormais faire référence à une expression d'alias immédiatement après l'avoir définie. Pour plus d'informations, consultez Liste SELECT.	18 juillet 2018
Spécification du type de compression lors de la création d'une table externe	Vous pouvez désormais spécifier le type de compression lors de la création d'une table externe avec Amazon Redshift Spectrum. Pour plus d'informations, consultez Création de tables externes.	27 juin 2018
PG_LAST_UNLOAD_ID	Ajout de la documentation relative à une nouvelle fonction d'informations système : PG_LAST_UNLOAD_ID. Pour plus d'informations, consultez PG_LAST_UNLOAD_ID.	27 juin 2018

ALTER TABLE RENAME COLUMN	ALTER TABLE prend désormais en charge le changement de nom de colonnes pour les tables externes. Pour plus d'informations, consultez Exemples de modification d'une table externe.	7 juin 2018

Mises à jour antérieures

Le tableau suivant décrit les modifications importantes apportées à chaque version du Guide du développeur de la base de données Amazon Redshift avant juin 2018.

Modification	Description	Date de modification
La commande COPY depuis Parquet inclut SMALLINT	La commande COPY prend désormais en charge le chargement de fichiers au format Parquet en colonnes qui utilisent le type de données SMALLINT. Pour de plus amples informations, veuillez consulter .	2 janvier 2019
COPY depuis les formats de données en colonnes	COPY prend désormais en charge le chargement à partir de fichiers sur Amazon S3 qui utilisent les formats de données en colonnes Parquet et ORC. Pour de plus amples informations, veuillez consulter .	17 mai 2018
Durée d'exécution maximale dynamique pour SQA	Par défaut, WLM attribue dynamiquement une valeur à l'exécution maximale SQA en fonction de l'analyse de la charge de travail de votre cluster. Pour de plus amples informations, veuillez consulter Durée d’exécution maximale pour les requêtes courtes.	17 mai 2018
Nouvelle colonne dans STL_LOAD_COMMITS	La table système STL_LOAD_COMMITS comporte une nouvelle colonne, `file_format` .	10 mai 2018

Modification	Description	Date de modification
Nouvelles colonnes de STL_HASHJOIN et autres tables des journaux systèmes	La table système STL_HASHJOIN comporte trois nouvelles colonnes, `hash_segment` , `hash_step` et `checksum`. De même, un `checksum` a été ajouté à STL_MERGEJOIN, STL_NESTLOOP, STL_HASH, STL_SCAN, STL_SORT, STL_LIMIT et STL_PROJECT.	17 mai 2018
Nouvelles colonnes dans STL_AGGR	La table système STL_AGGR se compose de deux nouvelles colonnes, `resizes` et `flushable` .	19 avril 2018
Nouvelles options pour les fonctions REGEX	Pour les fonctions REGEXP_INSTR et REGEXP_SUBSTR, vous pouvez à présent spécifier quel occurrence d'une correspondance utiliser et si la correspondance doit être sensible à la casse. REGEXP_INSTR vous permet aussi de spécifier s'il faut retourner la position du premier caractère de la correspondance ou celle du premier caractère après la fin de la correspondance.	22 mars 2018
Nouvelles colonnes dans les tables système	Les colonnes tombstonedblocks, tossedblocks et batched_by columns ont été ajoutées à la table système STL_COMMIT_STATS. La colonne localslice a été ajoutée à la vue système STV_SLICES.	22 mars 2018
Ajout et suppression de colonnes dans les tables externes	ALTER TABLE prend désormais en charge ADD COLUMN et DROP COLUMN pour les tables externes Amazon Redshift Spectrum.	22 mars 2018
Redshift Spectrum : nouvelles régions AWS	Redshift Spectrum est désormais disponible dans les régions de Mumbai et São Paulo. Pour obtenir une liste des régions prises en charge, consultez Régions Amazon Redshift Spectrum.	22 mars 2018

Modification	Description	Date de modification
La limite de table est passée à 20 000	Le nombre maximal de tables est maintenant de 20 000 pour les nœuds de cluster de type 8xlarge. La limite pour les nœuds de types large et xlarge est de 9 900. Pour de plus amples informations, veuillez consulter Limites et quotas.	13 mars 2018
Prise en charge de Redshift Spectrum pour JSON et Ion	Avec Redshift Spectrum, vous pouvez référencer des fichiers avec des données scalaires aux formats de données JSON ou Ion. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL TABLE.	26 février 2018
Chaînage de rôles IAM pour Redshift Spectrum	Vous pouvez enchaîner les rôles AWS Identity and Access Management (IAM) afin que votre cluster puisse assumer d'autres rôles non attachés au cluster, y compris des rôles appartenant à un autre AWS compte. Pour de plus amples informations, veuillez consulter Créer des rôles IAM dans Amazon Redshift Spectrum.	1 février 2018
ADD PARTITION prend en charge IF NOT EXISTS	La clause ADD PARTITION pour ALTER TABLE prend désormais en charge une option IF NOT EXISTS. Pour de plus amples informations, veuillez consulter ALTER TABLE.	11 janvier 2018
Données DATE pour les tables externes	Les tables externes Redshift Spectrum prennent désormais en charge le type de données DATE. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL TABLE.	11 janvier 2018
Redshift Spectrum : nouvelles régions AWS	Redshift Spectrum est désormais disponible dans les régions de Singapour, Sydney, Séoul et Francfort . Pour obtenir la liste des AWS régions prises en charge, consultezRégions Amazon Redshift Spectrum.	16 novembre 2017

Modification	Description	Date de modification
Accélération des requêtes courtes dans la gestion des charges de travail (WLM) d'Amazon Redshift	L'accélération des requêtes courtes (SQA) établit la priorité des requêtes de courte durée sélectionnées sur les requêtes de longue durée. La SQA exécute des requêtes de courte durée dans un espace dédié, afin que les requêtes SQA ne soient pas forcées d'attendre dans des files d'attente derrière les requêtes de longue durée. Avec la SQA, les requêtes de courte durée commencent à s'exécuter plus rapidement et les utilisateurs obtiennent les résultats plus rapidement. Pour de plus amples informations, veuillez consulter Accélération des requêtes courtes.	16 novembre 2017
WLM réaffecte les requêtes replacées	Au lieu d'annuler et de relancer une requête sautée, la gestion de la charge de travail (WLM) d'Amazon Redshift réaffecte désormais les requêtes admissibles à une nouvelle file d'attente. Lorsque WLM réaffecte une requête, elle déplace la requête dans la nouvelle file d'attente et continue l'exécution, ce qui permet de gagner du temps et d'économiser des ressources système. Les requêtes replacées qui ne peuvent pas être réaffectées sont redémarrées ou annulées. Pour de plus amples informations, veuillez consulter Saut de file d'attente des requêtes WLM.	16 novembre 2017
Accès aux journaux système pour les utilisateurs	Dans la plupart des tableaux de journaux système visibles des utilisateurs, les lignes générées par un autre utilisateur sont invisibles pour un utilisateur standard par défaut. Pour autoriser un utilisateur standard à voir toutes les lignes des tableaux visibles des utilisateurs, y compris les lignes générées par un autre utilisateur, exécutez ALTER USER ou CREATE USER et définissez le paramètre SYSLOG ACCESS sur UNRESTRICTED.	16 novembre 2017

Modification	Description	Date de modification
Mise en cache du résultat	Avec Mise en cache du résultat, lorsque vous exécutez une requête, Amazon Redshift met en cache le résultat. Lorsque vous exécutez à nouveau la requête, Amazon Redshift recherche une copie valide mise en cache du résultat de la requête. Si une correspondance est trouvée dans le cache des résultats, Amazon Redshift utilise le résultat mis en cache et n'exécute pas la requête. La mise en cache du résultat est activée par défaut. Pour désactiver la mise en cache du résultat, définissez le paramètre de configuration enable_result_cache_for_session sur `off`.	16 novembre 2017
Fonctions de métadonnées de colonne	PG_GET_COLS et PG_GET_LATE_BINDING_VIEW_COLS renvoient les métadonnées des colonnes pour les tables, les vues et les vues à liaison tardive d'Amazon Redshift.	16 novembre 2017
Saut de file d'attente WLM pour CTAS	La gestion des charges de travail (WLM) d'Amazon Redshift prend désormais en charge le saut de file d'attente des requêtes pour les déclarations CREATE TABLE AS (CTAS) ainsi que les requêtes en lecture seule, telles que les déclarations SELECT. Pour de plus amples informations, veuillez consulter Saut de file d'attente des requêtes WLM.	19 octobre 2017
Fichiers manifestes Amazon Redshift Sp	Lorsque vous créez une table externe Redshift Spectrum, vous pouvez spécifier un fichier manifeste qui répertorie les emplacements des fichiers de données sur Amazon S3. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL TABLE.	19 octobre 2017

Modification	Description	Date de modification
Amazon Redshift Spectrum AWS : nouvelles régions	Redshift Spectrum est désormais disponible dans les régions UE (Irlande) et Asie-Pacifique (Tokyo). Pour obtenir la liste des AWS régions prises en charge, consultezLimites d'Amazon Redshift Spectrum.	19 octobre 2017
Amazon Redshift Sp a ajouté des formats de fichiers	Vous pouvez désormais créer des tables externes Redshift Spectrum basées sur les formats de fichier de données Regex, OpenCSV et Avro. Pour de plus amples informations, veuillez consulter CREATE EXTERNAL TABLE.	5 octobre 2017
Pseudocolonnes pour tables externes Amazon Redshift Spectrum	Vous pouvez sélectionner les pseudo-colonnes `$path` et `$size` dans une table externe Redshift Spectrum pour afficher l'emplacement et la taille des fichiers de données référencés dans Amazon S3. Pour de plus amples informations, veuillez consulter Pseudocolonnes.	5 octobre 2017
Fonctions de validation JSON	Vous pouvez utiliser les fonctions IS_VALID_JSON et IS_VALID_JSON_ARRAY pour vérifier le formatage JSON. Les autres fonctions JSON disposent désormais d'un argument `null_if_invalid` facultatif.	5 octobre 2017
LISTAGG DISTINCT	Vous pouvez utiliser la clause DISTINCT avec la fonction d'agrégation LISTAGG et la fonction de fenêtre LISTAGG pour éliminer les valeurs en double dans l'expression spécifiée avant de procéder à la concaténation.	5 octobre 2017
Affichage des noms de colonnes en majuscules	Pour afficher les noms de colonnes en majuscules dans les résultats SELECT, vous pouvez définir le paramètre de configuration describe_field_name_in_uppercase sur `true`.	5 octobre 2017

Modification	Description	Date de modification
Ignorer les lignes d'en-tête dans les tables externes	Vous pouvez définir la propriété `skip.header.line.count` de la commande CREATE EXTERNAL TABLE de manière à ignorer les lignes d'en-tête au début des fichiers de données Redshift Spectrum.	5 octobre 2017
Nombre de lignes d’analyse	Les règles de contrôle des requêtes WLM utilisent la métrique scan_row_count pour retourner le nombre de lignes d'une étape d'analyse. Le nombre de lignes correspond au nombre total de lignes émises avant le filtrage des lignes marquées pour la suppression (lignes fantôme) et avant l’application des filtres de requête définis par l’utilisateur. Pour de plus amples informations, veuillez consulter Métriques de surveillance des requêtes pour cluster Amazon Redshift provisionné.	21 septembre 2017
Fonctions SQL définies par l'utilisateur	Une fonction scalaire SQL définie par l'utilisateur intègre une clause SELECT SQL qui s'exécute lorsque la fonction est appelée et renvoie une valeur unique. Pour de plus amples informations, veuillez consulter SQL scalaire UDFs.	31 août 2017

Modification	Description	Date de modification
Vues à liaison tardive	Une vue à liaison tardive n'est pas liée aux objets de base de données sous-jacents, tels que les tables et les fonctions définies par l'utilisateur. Par conséquent, il n'y a aucune dépendance entre la vue et les objets auxquels elle fait référence. Vous pouvez créer une vue même si les objets référencés n'existent pas. Parce qu'il n'y a aucune dépendance, vous pouvez supprimer ou modifier un objet référencé sans affecter la vue. Amazon Redshift ne vérifie pas les dépendances tant que la vue n'est pas interrogée. Pour créer une vue à liaison tardive, spécifiez la clause WITH NO SCHEMA BINDING avec l'instruction CREATE VIEW. Pour de plus amples informations, veuillez consulter CREATE VIEW.	31 août 2017
Fonction OCTET_LENGTH	OCTET_LENGTH renvoie la longueur de la chaîne spécifiée en tant que nombre d'octets.	18 août 2017
Types de fichier ORC et Grok pris en charge	Amazon Redshift Spectrum prend désormais en charge les formats de données ORC et Grok pour les fichiers de données Redshift Spectrum. Pour de plus amples informations, veuillez consulter Fichiers de données pour les requêtes dans Amazon Redshift Spectrum.	18 août 2017
RegexSerDe désormais pris en charge	Amazon Redshift Spectrum prend désormais en charge RegexSerDe le format de données. Pour de plus amples informations, veuillez consulter Fichiers de données pour les requêtes dans Amazon Redshift Spectrum.	19 juillet 2017

Modification	Description	Date de modification
Nouvelles colonnes ajoutées à SVV_TABLES et SVV_COLUMNS	Les colonnes `domain_name` et `remarks` ont été ajoutées à SVV_COLUMNS. Une colonne de remarques a été ajoutée à SVV_TABLES.	19 juillet 2017
Vues système SVV_TABLES et SVV_COLUMNS	Les vues système SVV_TABLES et SVV_COLUMNS fournissent des informations sur les colonnes et d'autres détails relatifs aux tables et aux vues locales et externes.	7 juillet 2017
VPC n'est plus nécessaire pour Amazon Redshift Spectrum avec le métastore Hive d'Amazon EMR	Redshift Spectrum a supprimé l'exigence selon laquelle le cluster Amazon Redshift et le cluster Amazon EMR doivent être dans le même VPC et le même sous-réseau lors de l'utilisation d'un métastore Amazon EMR Hive. Pour de plus amples informations, veuillez consulter Utiliser des catalogues externes dans Amazon Redshift Spectrum.	7 juillet 2017
Exécution de la commande UNLOAD vers des fichiers de taille inférieure	Par défaut, UNLOAD crée plusieurs fichiers sur Amazon S3 d'une taille maximale de 6,2 Go. Pour créer des fichiers plus petits, spécifiez MAXFILESIZE avec la commande UNLOAD. Vous pouvez spécifier une taille de fichier maximale située entre 5 Mo et 6,2 Go. Pour de plus amples informations, veuillez consulter UNLOAD.	7 juillet 2017
TABLE PROPERTIES	Vous pouvez maintenant définir le paramètre numRows de TABLE PROPERTIES pour CREATE EXTERNAL TABLE ou ALTER TABLE afin de mettre à jour les statistiques de table pour refléter le nombre de lignes dans la table.	6 juin 2017

Modification	Description	Date de modification
ANALYZE PREDICATE COLUMNS	Pour gagner du temps et économiser des ressources de cluster, vous pouvez choisir d'analyser uniquement les colonnes susceptibles d'être utilisées comme prédicats. Lorsque vous exécutez ANALYZE avec la clause PREDICATE COLUMNS, l'opération d'analyse inclut uniquement les colonnes qui ont été utilisées dans une jointure, une condition de filtre ou une clause group by, ou qui sont utilisées en tant que clé de tri ou clé de distribution. Pour de plus amples informations, veuillez consulter Analyse des tables.	25 mai 2017
Stratégies IAM pour Amazon Redshift Spectrum	Pour accorder l'accès à un compartiment Amazon S3 en n'utilisant que Redshift Spectrum, vous pouvez inclure une condition qui autorise l'accès pour l'agent utilisateur `AWS Redshift/Spectrum` . Pour de plus amples informations, veuillez consulter Politiques IAM pour Amazon Redshift Spectrum.	25 mai 2017
Analyse récursive d'Amazon Redshift S	Redshift Spectrum analyse désormais les fichiers dans les sous-dossiers ainsi que le dossier spécifié dans Amazon S3. Pour de plus amples informations, veuillez consulter Tables externes pour Redshift Spectrum.	25 mai 2017
Règles de surveillance de requête	À l'aide des règles de surveillance des requêtes WLM, vous pouvez définir des limites de performance basées sur des métriques pour les files d'attente WLM et spécifier l'action à entreprendre lorsqu'une requête dépasse ces limites — log, hop ou abort. Vous définissez les règles de surveillance de requête dans le cadre de la configuration de la gestion de la charge de travail (WLM). Pour de plus amples informations, veuillez consulter Règles de surveillance de requête WLM.	21 avril 2017

Modification	Description	Date de modification
Amazon Redshift Sp	Grâce à Redshift Spectrum, vous pouvez interroger et récupérer efficacement des données à partir de fichiers dans Amazon S3 sans avoir à charger les données dans des tables. Les requêtes Redshift Spectrum s'exécutent très rapidement contre les grands ensembles de données car Redshift Spectrum analyse les fichiers de données directement dans Amazon S3. Une grande partie du traitement s'effectue dans la couche Amazon Redshift Spectrum, et la plupart des données restent dans Amazon S3. Plusieurs clusters peuvent interroger simultanément le même jeu de données sur Amazon S3 sans qu'il soit nécessaire de faire des copies des données pour chaque cluster. Pour de plus amples informations, veuillez consulter .	19 avril 2017
Nouvelles tables système pour la prise en charge de Redshift Spectrum	Les nouvelles vues système suivantes ont été ajoutées afin de prendre en charge Redshift Spectrum : • SVL_S3QUERY • SVL_S3QUERY_SUMMARY • SVV_EXTERNAL_COLUMNS • SVV_EXTERNAL_DATABASES • SVV_EXTERNAL_PARTITIONS • SVV_EXTERNAL_TABLES • PG_EXTERNAL_SCHEMA	19 avril 2017

Modification	Description	Date de modification
Fonction d'agrégation APPROXIMATE PERCENTILE_DISC	La fonction d'agrégation APPROXIMATE PERCENTILE_DISC est désormais disponible.	4 avril 2017
Chiffrement côté serveur avec KMS	Vous pouvez désormais décharger des données vers Amazon S3 en utilisant le chiffrement côté serveur avec une clé AWS Key Management Service (SSE-KMS). En outre, COPY charge désormais de manière transparente les fichiers de données chiffrées par KMS depuis Amazon S3. Pour de plus amples informations, veuillez consulter UNLOAD.	9 février 2017
Nouvelle syntaxe d'autorisation	Vous pouvez désormais utiliser les paramètres IAM_ROLE, MASTER_SYMMETRIC_KEY, ACCESS_KEY_ID, SECRET_ACCESS_KEY et SESSION_TOKEN pour fournir des informations d'autorisation et d'accès pour les commandes COPY, UNLOAD et CREATE LIBRARY. La nouvelle syntaxe d'autorisation constitue une façon plus flexible de fournir un argument de chaîne au paramètre CREDENTIALS. Pour de plus amples informations, veuillez consulter Paramètres d’autorisation.	9 février 2017
Augmentation de la limite des schémas	Vous pouvez désormais créer jusqu'à 9 900 schémas par cluster. Pour de plus amples informations, veuillez consulter CREATE SCHEMA.	9 février 2017

Modification	Description	Date de modification
Encodage de table par défaut	CREATE TABLE et ALTER TABLE attribuent une compression LZO à la plupart des nouvelles colonnes. Les colonnes définies comme des clés de tri avec le type de données BOOLEAN, REAL ou DOUBLE PRECISION et les tables temporaires se voient attribuer une compression RAW par défaut. Pour de plus amples informations, veuillez consulter ENCODE.	6 février 2017
Encodage de compression ZSTD	Amazon Redshift prend désormais en charge le codage par compression de colonne ZSTD.	19 janvier 2017
Fonctions d'agrégation PERCENTILE_CONT et MEDIAN	PERCENTILE_CONT et MEDIAN sont désormais disponibles en tant que fonctions d'agrégation et fonctions de fenêtrage.	19 janvier 2017
Journalisation utilisateur UDF (fonction définie par l'utilisateur)	Vous pouvez utiliser le module de journalisation Python pour créer des messages d'erreur et d'avertissement définis par l'utilisateur dans votre UDFs. Après l'exécution d'une requête, vous pouvez interroger la vue système SVL_UDF_LOG pour récupérer les messages journalisés. Pour plus d'informations sur les messages définis par l'utilisateur, consultez Erreurs de journalisation et avertissements en Python UDFs	8 décembre 2016
Estimation de la réduction avec ANALYZE COMPRESSION	La commande ANALYZE COMPRESSION indique maintenant une estimation pour le pourcentage de réduction de l'espace sur le disque pour chaque colonne. Pour de plus amples informations, veuillez consulter ANALYZE COMPRESSION.	10 novembre 2016

Modification	Description	Date de modification
Limites de connexions	Vous pouvez maintenant définir une limite pour le nombre de connexions à la base de données qu'un utilisateur est autorisé à ouvrir simultanément. Vous pouvez également limiter le nombre de connexions simultanées pour une base de données. Pour plus d'informations, consultez CREATE USER et CREATE DATABASE.	10 novembre 2016
Amélioration de l'ordre de tri avec COPY	La commande COPY ajoute maintenant automatiquement des lignes nouvelles à la région triée de la table lorsque vous chargez vos données dans l'ordre des clés de tri. Pour connaître les conditions d'activation de cette amélioration, consultez Chargez vos données dans l'ordre des clés de tri	10 novembre 2016
CTAS avec compression	CREATE TABLE AS (CTAS) attribue désormais automatiquement les encodages de compression aux nouvelles tables en fonction du type de données de la colonne. Pour de plus amples informations, veuillez consulter Héritage des attributs de colonne et de table.	28 octobre 2016
Type de données d'horodatage avec fuseau horaire	Amazon Redshift prend désormais en charge un type de données timestamp avec fuseau horaire (TIMESTAMPTZ). En outre, plusieurs nouvelles fonctions ont été ajoutées pour prendre en charge le nouveau type de données. Pour de plus amples informations, veuillez consulter Fonctions de date et d'heure.	29 septembre 2016

Modification	Description	Date de modification
Seuil d’analyse	Pour réduire le temps de traitement et améliorer les performances globales du système des opérations ANALYSE, Amazon Redshift ignore l'analyse d'une table si le pourcentage de lignes qui ont été modifiées depuis l'exécution de la dernière commande ANALYZE est inférieur au seuil d'analyse spécifié par le paramètre analyze_threshold_percent. Par défaut, `analyze_threshold_percent` est 10.	9 août 2016
Nouvelle table système STL_RESTARTED_SESSIONS	Lorsque Amazon Redshift redémarre une session, STL_RESTARTED_SESSIONS enregistre le nouvel identifiant de processus (PID) et l'ancien PID.	9 août 2016
Mise à jour de la documentation Fonctions Date et heure	Ajout d'un résumé des fonctions avec des liens vers la rubrique Fonctions de date et d’heure et mise à jour des références de la fonction à des fins de cohérence.	24 juin 2016
Nouvelles colonnes dans STL_CONNECTION_LOG	La table système STL_CONNECTION_LOG comporte deux nouvelles colonnes pour suivre les connexions SSL. Si vous chargez régulièrement les journaux d’audit dans une table Amazon Redshift, vous devrez ajouter les nouvelles colonnes suivantes à la table cible : sslcompression et sslexpansion.	5 mai 2016
MD5-mot de passe de hachage	Vous pouvez spécifier le mot de passe d'une ALTER USER commande CREATE USER or en fournissant la chaîne MD5 -hash du mot de passe et du nom d'utilisateur.	21 avril 2016
Nouvelle colonne dans STV_TBL_PERM	La colonne `backup` dans la vue système STV_TBL_PERM indique si la table est incluse dans les instantanés du cluster. Pour de plus amples informations, veuillez consulter BACKUP.	21 avril 2016

Modification	Description	Date de modification
Tables sans sauvegarde	Pour les tables, telles que les tables intermédiaires, qui ne contiennent pas de données critiques, vous pouvez spécifier BACKUP NO dans votre instruction CREATE TABLE ou CREATE TABLE AS afin d'empêcher Amazon Redshift d'inclure la table dans les instantanés automatiques ou manuels. L'utilisation de tables sans sauvegarde permet d'économiser du temps lors de la création d'instantanés et de la restauration à partir d'instantanés et réduit l'espace de stockage sur Amazon S3.	7 avril 2016
Seuil de suppression de VACUUM	Par défaut, la commande VACUUM permet maintenant de récupérer de l'espace, tel qu'au moins 95 % des lignes restantes ne sont pas marquées pour suppression. Par conséquent, la phase de suppression de la commande VACUUM est généralement beaucoup plus rapide que la phase de récupération de 100 % des lignes supprimées. Vous pouvez changer le seuil par défaut pour une seule table en incluant le paramètre TO threshold PERCENT lorsque vous exécutez la commande VACUUM.	7 avril 2016
Table système SVV_TRANSACTIONS	La vue système SVV_TRANSACTIONS enregistre des informations sur les transactions qui maintiennent actuellement les verrous dans les tables de la base de données.	7 avril 2016

Modification	Description	Date de modification
Utilisation des rôles IAM pour accéder à d'autres ressources AWS	Pour déplacer des données entre votre cluster et une autre AWS ressource, telle qu'Amazon S3, DynamoDB, Amazon EMR ou EC2 Amazon, votre cluster doit être autorisé à accéder à la ressource et à effectuer les actions nécessaires. Comme alternative plus sécurisée pour fournir une paire de clés d'accès avec des commandes COPY, UNLOAD ou CREATE LIBRARY, vous pouvez maintenant spécifier un rôle IAM que votre cluster utilise pour l'authentification et l'autorisation. Pour de plus amples informations, veuillez consulter Contrôle d’accès basé sur les rôles.	29 mars 2016
Seuil de tri VACUUM	La commande VACUUM ignore maintenant la phase de tri des tables dans lesquelles plus de 95 % des lignes de la table sont déjà triées. Vous pouvez changer le seuil de tri par défaut pour une seule table en incluant le paramètre TO threshold PERCENT lorsque vous exécutez la commande VACUUM.	17 mars 2016
Nouvelles colonnes dans STL_CONNECTION_LOG	La table système STL_CONNECTION_LOG contient trois nouvelles colonnes. Si vous chargez régulièrement des journaux d'audit dans une table Amazon Redshift, vous devrez ajouter les nouvelles colonnes suivantes à la table cible : sslversion, sslcipher et mtu.	17 mars 2016
UNLOAD avec la compression bzip2	Vous avez maintenant la possibilité d'exécuter la commande UNLOAD à l'aide de la compression bzip2.	8 février 2016
ALTER TABLE APPEND	ALTER TABLE APPEND ajoute des lignes à une table cible en déplaçant les données à partir d'une table source existante. ALTER TABLE APPEND est généralement beaucoup plus rapide qu’une opération CREATE TABLE AS ou INSERT INTO semblable, car les données sont déplacées, pas dupliquées.	8 février 2016

Modification	Description	Date de modification
Saut de file d'attente des requêtes WLM	Si la gestion de la charge de travail (WLM) annule une requête en lecture seule, telle qu'une instruction SELECT, en raison d'un délai WLM, WLM tente d'acheminer la requête vers la prochaine file d'attente correspondante. Pour de plus amples informations, veuillez consulter .	7 janvier 2016
ALTER DEFAULT PRIVILEGES	Vous pouvez utiliser la commande ALTER DEFAULT PRIVILEGES pour définir le groupe de privilèges d'accès par défaut sur des objets qui seront créés à l'avenir par l'utilisateur spécifié.	10 décembre 2015
Compression d'un fichier bzip2	La commande COPY prend en charge le chargement de données à partir de fichiers qui ont été compressés à l'aide de bzip2.	10 décembre 2015
NULLS FIRST et NULLS LAST	Vous pouvez spécifier si une clause ORDER BY doit classer les valeurs NULLS en premier ou en dernier dans l'ensemble de résultats. Pour plus d'informations, consultez Clause ORDER BY et Récapitulatif de la syntaxe de la fonction de fenêtrage.	19 novembre 2015
Mot-clé REGION pour CREATE LIBRARY	Si le compartiment Amazon S3 qui contient les fichiers de bibliothèque UDF ne réside pas dans la même AWS région que votre cluster Amazon Redshift, vous pouvez utiliser l'option REGION pour spécifier la région dans laquelle se trouvent les données. Pour de plus amples informations, veuillez consulter CREATE LIBRARY.	19 novembre 2015

Modification	Description	Date de modification
Fonctions scalaires définies par l'utilisateur () UDFs	Vous pouvez désormais créer des fonctions scalaires personnalisées définies par l'utilisateur pour implémenter des fonctionnalités de traitement non SQL fournies soit par les modules pris en charge par Amazon Redshift dans la bibliothèque standard Python 2.7, soit par vos propres fonctions personnalisées UDFs basées sur le langage de programmation Python. Pour de plus amples informations, veuillez consulter Fonctions définies par l'utilisateur dans Amazon Redshift.	11 septembre 2015
Propriétés dynamiques de la configuration WLM	Le paramètre de configuration WLM prend désormais en charge l'application dynamique de certaines propriétés. D'autres propriétés restent statiques et nécessitent que les clusters associés soient redémarrés afin que les modifications de configuration puissent être appliquées. Pour plus d'informations, consultez Propriétés de configuration dynamiques et statiques WLM et Gestion de la charge de travail.	3 août 2015
Fonction LISTAGG	Les fonctions Fonction LISTAGG et Fonction de fenêtrage LISTAGG renvoient une chaîne créée en concaténant un ensemble de valeurs de la colonne.	30 juillet 2015
Paramètre obsolète	Le paramètre de configuration max_cursor_result_set_size est obsolète. La taille des ensembles de résultats de curseur est restreinte en fonction du type de nœud du cluster. Pour de plus amples informations, veuillez consulter Contraintes de curseur.	24 juillet 2015
Documentation de commande COPY révisée	La référence de commande COPY a été considérablement révisée pour rendre le matériel plus convivial et plus accessible.	15 juillet 2015

Modification	Description	Date de modification
Exécuter la commande COPY depuis le format Avro	La commande COPY prend en charge le chargement de données au format Avro à partir de fichiers de données sur Amazon S3, Amazon EMR, et à partir d'hôtes distants via SSH. Pour plus d'informations, consultez AVRO et Copier depuis des exemples Avro.	8 juillet 2015
STV_STARTUP_RECOVERY_STATE	La table système STV_STARTUP_RECOVERY_STATE enregistre l'état des tables qui sont temporairement verrouillées pendant les opérations de redémarrage du cluster. Amazon Redshift place un verrou temporaire sur les tables pendant qu'elles sont traitées pour résoudre les transactions devenues obsolètes à la suite d'un redémarrage du cluster.	25 mai 2015
Clause ORDER BY facultative pour classer les fonctions	La clause ORDER BY est désormais facultative pour certaines fonctions de rang de fenêtre. Pour plus d'informations, consultez Fonction de fenêtrage CUME_DIST, Fonction de fenêtrage DENSE_RANK, Fonction de fenêtrage RANK, Fonction de fenêtrage NTILE, Fonction de fenêtrage PERCENT_RANK et Fonction de fenêtrage ROW_NUMBER.	25 mai 2015
Clés de tri entrelacées	Les clés de tri entrelacées donnent un poids égal à chaque colonne dans la clé de tri. L'utilisation des clés de tri entrelacées au lieu des clés composées par défaut améliore nettement les performances des requêtes qui utilisent des prédicats restrictifs sur les colonnes de tri secondaire, surtout pour les grandes tables. Le tri entrelacé améliore également la performance globale lorsque plusieurs requêtes filtrent différentes colonnes dans la même table. Pour plus d'informations, consultez Clés de tri et CREATE TABLE.	le 11 mai 2015

Modification	Description	Date de modification
Rubrique Réglage de performances des requêtes révisée	La rubrique Réglage des performances des requêtes a été étendue afin d'inclure de nouvelles requêtes pour analyser les performances des requêtes et d'autres exemples. La rubrique a également été révisée de manière à être plus claire et plus complète. Bonnes pratiques Amazon Redshift pour la conception de requêtes contient des informations supplémentaires sur la façon d'écrire des requêtes pour améliorer les performances.	23 mars 2015
SVL_QUERY_QUEUE_INFO	La vue SVL_QUERY_QUEUE_INFO résume les détails des requêtes qui ont passé du temps dans une file d'attente de requête WLM ou une file d'attente de validation.	le 19 février 2015
SVV_TABLE_INFO	Vous pouvez utiliser la vue SVV_TABLE_INFO pour diagnostiquer et traiter les problèmes de conception de table qui peuvent influencer les performances des requêtes, y compris les problèmes d'encodage de compression, les clés de distribution, le style de tri, l'asymétrie de la distribution des données, la taille de la table et les statistiques.	le 19 février 2015
La commande UNLOAD utilise le chiffrement de fichier côté serveur	La commande UNLOAD utilise désormais automatiquement le chiffrement côté serveur (SSE) d'Amazon S3 pour chiffrer tous les fichiers de données de déchargement. Le chiffrement côté serveur ajoute une autre couche de sécurité de données avec peu ou pas de modification de performances.	31 octobre 2014
Fonction de fenêtrage CUME_DIST	La fonction Fonction de fenêtrage CUME_DIST calcule la distribution cumulée d'une valeur au sein d'une fenêtre ou une partition.	31 octobre 2014

Modification	Description	Date de modification
Fonction MONTHS_BETWEEN	La fonction Fonction MONTHS_BETWEEN détermine le nombre de mois entre deux dates.	31 octobre 2014
Fonction NEXT_DAY	La fonction Fonction NEXT_DAY renvoie la date de la première instance d'une date spécifiée qui est ultérieure à la date donnée.	31 octobre 2014
Fonction de fenêtrage PERCENT_RANK	La fonction Fonction de fenêtrage PERCENT_RANK calcule le rang en pourcentage d'une ligne donnée.	31 octobre 2014
Fonction de fenêtrage RATIO_TO_REPORT	La fonction Fonction de fenêtrage RATIO_TO_REPORT calcule le ratio d'une valeur par rapport à la somme des valeurs dans une fenêtre ou une partition.	31 octobre 2014
Fonction TRANSLATE	La fonction Fonction TRANSLATE remplace toutes les occurrences de caractères spécifiés au sein d'une expression donnée par des produits de remplacement spécifiés.	31 octobre 2014
NVL2 fonction	La fonction NVL2 fonction renvoie l'une des deux valeurs selon qu'une expression spécifique a une valeur NULL ou NOT NULL.	16 octobre 2014
Fonction de fenêtrage MEDIAN	La fonction Fonction de fenêtrage MEDIAN calcule la valeur médiane de la plage de valeurs dans une fenêtre ou une partition.	16 octobre 2014

Modification	Description	Date de modification
Clause ON ALL TABLES IN SCHEMA schema_name pour les commandes GRANT et REVOKE	Les commandes GRANT et REVOKE ont été mises à jour avec une clause ON ALL TABLES IN SCHEMA schema_name. Cette clause vous permet d'utiliser une simple commande pour modifier les privilèges de toutes les tables dans un schéma.	16 octobre 2014
Clause IF EXISTS pour les commandes DROP SCHEMA, DROP TABLE, DROP USER, et DROP VIEW	Les commandes DROP SCHEMA, DROP TABLE, DROP USER, et DROP VIEW ont été mises à jour avec une clause IF EXISTS. Cette clause empêche la commande de faire des modifications et renvoie un message plutôt que de s'arrêter avec une erreur si l'objet spécifié n'existe pas.	16 octobre 2014
Clause IF NOT EXISTS pour les commandes CREATE SCHEMA et CREATE TABLE	Les commandes CREATE SCHEMA et CREATE TABLE ont été mises à jour avec une clause IF NOT EXISTS. Cette clause empêche la commande de faire des modifications et renvoie un message plutôt que de s'arrêter avec une erreur si l'objet spécifié existe déjà.	16 octobre 2014
Prise en charge de la commande COPY pour l'encodage UTF-16	La commande COPY prend désormais en charge le chargement de fichiers de données qui utilisent l'encodage UTF-16, ainsi que l'encodage UTF-8. Pour de plus amples informations, veuillez consulter ENCODING.	29 septembre 2014

Modification	Description	Date de modification
Nouveau didacticiel Gestion de la charge de travail	Didacticiel : Configuration des files d'attente de gestion manuelle de la charge de travail vous guide à travers le processus de configuration des files d'attente de gestion de la charge de travail (WLM) pour améliorer le traitement des requêtes et l'allocation des ressources.	25 septembre 2014
Chiffrement AES 128 bits.	La commande COPY prend désormais en charge le chiffrement AES 128 bits ainsi que le chiffrement AES 256 bits lors du chargement à partir de fichiers de données chiffrées à l'aide du chiffrement côté client d'Amazon S3. Pour de plus amples informations, veuillez consulter Chargement de fichiers de données chiffrés à partir d'Amazon S3.	29 septembre 2014
Fonction PG_LAST_UNLOAD_COUNT	La fonction PG_LAST_UNLOAD_COUNT renvoie le nombre de lignes qui ont été traitées lors de l'opération UNLOAD la plus récente. Pour de plus amples informations, veuillez consulter PG_LAST_UNLOAD_COUNT.	15 septembre 2014
Nouvelle section Dépannage des requêtes	Résolution des problèmes de requêtes fournit une référence rapide pour identifier et résoudre certains des problèmes les plus courants et les plus graves que vous êtes susceptibles de rencontrer avec les requêtes Amazon Redshift.	le 7 juillet 2014
Nouveau didacticiel Chargement des données	Didacticiel : chargement des données à partir d'Amazon S3 vous guide du début à la fin du processus de chargement de données dans vos tables de base de données Amazon Redshift à partir de fichiers de données dans un compartiment Amazon S3.	1 juillet 2014

Modification	Description	Date de modification
Fonction de fenêtrage PERCENTILE_CONT	La fonction Fonction de fenêtrage PERCENTILE_CONT est une fonction de distribution inverse qui suppose un modèle de distribution continue. Elle prend une valeur de centile et une spécification de tri, et renvoie une valeur interpolée qui entre dans la catégorie de la valeur de centile donnée en ce qui concerne la spécification de tri.	30 juin 2014
Fonction de fenêtrage PERCENTILE_DISC	La fonction Fonction de fenêtrage PERCENTILE_DISC est une fonction de distribution inverse qui suppose un modèle de distribution discrète. Elle prend une valeur de centile et une spécification de tri et renvoie un élément de l'ensemble.	30 juin 2014
Fonctions GREATEST et LEAST	Les fonctions Fonctions GREATEST et LEAST renvoient la valeur plus grande ou la plus petite à partir d'une liste d'expressions.	30 juin 2014
Exécution de la commande COPY entre régions	La commande COPY prend en charge le chargement de données à partir d'un compartiment Amazon S3 ou d'une table Amazon DynamoDB qui se trouve dans une région différente de celle du cluster Amazon Redshift. Pour plus d’informations, consultez REGION dans la référence de la commande COPY.	30 juin 2014
Bonnes pratiques étendues	La rubrique Bonnes pratiques Amazon Redshift a été élargie, réorganisée et déplacée vers le haut de la hiérarchie de navigation pour la rendre plus visible.	28 mai 2014

Modification	Description	Date de modification
Exécution de la commande UNLOAD vers un seul fichier	La commande UNLOAD peut éventuellement décharger les données de la table en série vers un seul fichier sur Amazon S3 en ajoutant l'option PARALLEL OFF. Si la taille des données est supérieure à la taille de fichier maximale de 6,2 Go, la commande UNLOAD crée des fichiers supplémentaires.	6 mai 2014
Fonctions REGEXP	Les fonctions REGEXP_COUNT, REGEXP_INSTR, et REGEXP_REPLACE manipulent des chaînes basées sur la mise en correspondance de modèles d'expressions régulières.	6 mai 2014
COPY depuis Amazon EMR	La commande COPY prend en charge le chargement de données directement à partir des clusters Amazon EMR. Pour de plus amples informations, veuillez consulter Chargement de données à partir d'Amazon EMR.	18 avril 2014
Augmentation de la limite de simultanéité WLM	Vous pouvez configurer maintenant la gestion de la charge de travail (WLM) pour exécuter jusqu'à 50 requêtes simultanément dans les files d'attente de requêtes définies par l'utilisateur. Cette augmentation offre aux utilisateurs plus de flexibilité pour la gestion des performances système en modifiant les configurations WLM. Pour de plus amples informations, veuillez consulter .	18 avril 2014

Modification	Description	Date de modification
Nouveau paramètre de configuration pour gérer la taille du curseur	Le paramètre de configuration `max_cursor_result_set_size` définit la taille maximale des données, en mégaoctets, qui peuvent être renvoyées par le jeu de résultats du curseur d'une requête plus grande. Cette valeur de paramètre affecte également le nombre de curseurs simultanés du cluster, ce qui vous permet de configurer une valeur qui augmente ou diminue le nombre de curseurs de votre cluster. Pour plus d'informations, veuillez consulter la rubrique DECLARE de ce guide et la rubrique Configurer la taille maximale d'un ensemble de résultats de curseur dans le Guide de gestion du cluster Amazon Redshift.	28 mars 2014
Exécution de la commande COPY à partir du format JSON	La commande COPY prend en charge le chargement de données au format JSON depuis des fichiers de données sur Amazon S3 et depuis des hôtes distants via SSH. Pour plus d'informations, consultez les notes d'utilisation de Exécution de la commande COPY à partir du format JSON.	25 mars 2014
Nouvelle table système STL_PLAN_INFO	La table STL_PLAN_INFO complète la commande EXPLAIN comme autre moyen d'examiner les plans de requête.	25 mars 2014
Nouvelle fonction REGEXP_SUBSTR	La fonction Fonction REGEXP_SUBSTR renvoie les caractères extraits d'une chaîne en recherchant un modèle d'expression régulière.	25 mars 2014
Nouvelles colonnes pour STL_COMMIT_STATS	La table STL_COMMIT_STATS se compose de deux nouvelles colonnes : `numxids` et `oldestxid` .	6 mars 2014

Modification	Description	Date de modification
Exécution de la commande COPY à partir de la prise en charge SSH pour gzip et lzop	La commande COPY prend en charge la compression gzip et lzop lors du chargement de données via une connexion SSH.	13 février 2014
Nouvelles fonctions	La fonction Fonction de fenêtrage ROW_NUMBER renvoie le numéro de la ligne en cours. La fonction Fonction STRTOL convertit une expression de chaîne d'un nombre de la base spécifiée en la valeur de nombre entier équivalente. Les fonctions PG_CANCEL_BACKEND et PG_TERMINATE_BACKEND permettent aux utilisateurs d'annuler des requêtes et des connexions de session. La fonction LAST_DAY a été ajoutée pour la compatibilité avec Oracle.	13 février 2014
Nouvelle table système	La table système STL_COMMIT_STATS fournit des métriques associées à la validation des performances, y compris la chronologie des différentes étapes de validation et le nombre de blocs engagés.	13 février 2014
Exécution de la commande FETCH avec des clusters à nœud unique	Lorsque vous utilisez un curseur sur un cluster à nœud unique, le nombre maximal de lignes pouvant être extraites à l'aide de la commande FETCH est de 1 000. La commande FETCH FORWARD ALL n'est pas prise en charge pour les clusters à nœud unique.	13 février 2014
Stratégie de redistribution DS_DIST_ALL_INNER	DS_DIST_ALL_INNER dans la sortie du plan Explain indique que la totalité de la table interne a été redistribuée à une seule tranche, car la table externe utilise DISTSTYLE ALL. Pour plus d'informations, consultez Exemples de types de jointures et Évaluation du plan de requête.	13 janvier 2014

Modification	Description	Date de modification
Nouvelles tables système pour les requêtes	Amazon Redshift a ajouté de nouvelles tables système que les clients peuvent utiliser pour évaluer l'exécution des requêtes à des fins de réglage et de dépannage . Pour plus d'informations, consultez SVL_COMPILE, STL_SCAN, STL_RETURN, STL_SAVE STL_ALERT _EVENT_LOG.	13 janvier 2014
Curseurs à nœud unique	Les curseurs sont maintenant pris en charge pour les clusters à nœud unique. Un cluster à nœud unique peut avoir deux curseurs ouverts à la fois, avec un ensemble de résultats maximal de 32 Go. Sur un cluster à nœud unique, nous vous recommandons de définir le paramètre de la taille du cache ODBC sur 1 000. Pour de plus amples informations, veuillez consulter DECLARE.	13 décembre 2013
Style de distribution ALL	La distribution ALL peut considérablement raccourcir les durées d'exécution de certains types de requêtes. Lorsqu'une table utilise le style de distribution ALL, une copie de la table est distribuée à chaque nœud. Étant donné que la table est effectivement colocalisée avec toutes les autres tables, aucune redistribution n'est nécessaire au cours de l'exécution des requêtes. La distribution ALL n'est pas appropriée pour toutes les tables, car elle augmente les besoins de stockage et le temps de chargement. Pour de plus amples informations, veuillez consulter Distribution des données pour l'optimisation des requêtes.	11 novembre 2013

Modification	Description	Date de modification
Exécution de la commande COPY depuis les hôtes distants	Outre le chargement de tables à partir de fichiers de données sur Amazon S3 et à partir de tables Amazon DynamoDB, la commande COPY peut charger des données texte à partir de clusters Amazon EMR, d'instances EC2 Amazon et d'autres hôtes distants à l'aide de connexions SSH. Amazon Redshift utilise plusieurs connexions SSH simultanées pour lire et charger des données en parallèle. Pour de plus amples informations, veuillez consulter Chargement des données à partir des hôtes distants.	11 novembre 2013
Pourcentage de la mémoire WLM utilisée	Vous pouvez équilibrer la charge de travail en désignant un pourcentage spécifique de la mémoire pour chaque file d'attente dans la configuration de votre gestion de la charge de travail (WLM). Pour de plus amples informations, veuillez consulter Implémentation de la gestion manuelle de la charge de travail.	11 novembre 2013
APPROXIMATE COUNT(DISTINCT)	Les requêtes qui utilisent APPROXIMATE COUNT(DISTINCT) s'exécutent beaucoup plus vite, avec une erreur relative de 2 %. La fonction APPROXIMATIVE COUNT (DISTINCT) utilise un HyperLogLog algorithme. Pour plus d'informations, consultez le Fonction COUNT.	11 novembre 2013
Nouvelles fonctions SQL permettant d'extraire les détails de la requête récente	Quatre nouvelles fonctions SQL récupèrent des détails sur les requêtes récentes et les commandes COPY. Les nouvelles fonctions facilitent l'interrogation des tables de journal système et dans de nombreux cas, fournissent des détails nécessaires sans avoir besoin d'accéder aux tables système. Pour plus d'informations, consultez PG_BACKEND_PID, PG_LAST_COPY_ID, PG_LAST_COPY_COUNT, PG_LAST_QUERY_ID.	1er novembre 2013

Modification	Description	Date de modification
Option MANIFEST pour UNLOAD	L'option MANIFEST pour la commande UNLOAD complète l'option MANIFEST pour la commande COPY. L'utilisation de l'option MANIFEST avec UNLOAD crée automatiquement un fichier manifeste qui liste explicitement les fichiers de données qui ont été créés sur Amazon S3 par l'opération de déchargement. Vous pouvez ensuite utiliser le même fichier manifeste avec une commande COPY pour charger les données. Pour plus d'informations, consultez Déchargement de données vers Amazon S3 et Exemples UNLOAD.	1er novembre 2013
Option MANIFEST pour la commande COPY	Vous pouvez utiliser l'option MANIFEST avec la commande COPY pour lister explicitement les fichiers de données qui seront chargés depuis Amazon S3.	18 octobre 2013
Tables système pour dépanner les requêtes	Ajout de documentation pour les tables système qui sont utilisées pour dépanner les requêtes. La section Vues STL pour la journalisation contient maintenant la documentation des tables système suivantes : STL_AGGR, STL_BCAST, STL_DIST, STL_DELETE, STL_HASH, STL_HASHJOIN, STL_INSERT, STL_LIMIT, STL_MERGE, STL_MERGEJOIN, STL_NESTLOOP, STL_PARSE, STL_PROJECT, STL_SCAN, STL_SORT, STL_UNIQUE, STL_WINDOW.	3 octobre 2013
Fonction CONVERT_TIMEZONE	La fonction Fonction CONVERT_TIMEZONE convertit un horodatage d'un fuseau horaire à un autre, avec la possibilité d'ajuster automatiquement l'heure d'été.	3 octobre 2013
Fonction SPLIT_PART	La fonction Fonction SPLIT_PART divise une chaîne sur le délimiteur spécifié et renvoie la partie à la position spécifiée.	3 octobre 2013

Modification	Description	Date de modification
Table système STL_USERLOG	STL_USERLOG permet d'enregistrer les détails des changements qui se produisent lorsqu'un utilisateur de base de données est créé, modifié ou supprimé.	3 octobre 2013
Encodage de colonne LZO et compression de fichiers LZOP.	Le codage par compression de colonne LZO associe un taux de compression très élevé à de bonnes performances. L'exécution de la commande COPY à partir de Amazon S3 prend en charge le chargement des fichiers compressés à l'aide de la compression LZOP.	19 septembre 2013
JSON, expressions régulières et curseurs	Prise en charge supplémentaire pour l'analyse des chaînes JSON, la mise en correspondance de modèles à l'aide d'expressions régulières et l'utilisation de curseurs pour récupérer des ensembles de données volumineux via une connexion ODBC. Pour plus d'informations, consultez Fonctions JSON, Conditions de correspondance de modèles et DECLARE.	10 septembre 2013
Option ACCEPTINVCHAR pour la commande COPY	Vous pouvez charger correctement les données qui contiennent des caractères UTF-8 valides en spécifiant l'option ACCEPTINVCHAR avec la commande COPY.	29 août 2013
Option CSV pour la commande COPY	La commande COPY prend maintenant en charge le chargement à partir des fichiers d'entrée au format CSV.	9 août 2013
CRC32	La fonction CRC32 fonction effectue des contrôles de redondance cyclique.	9 août 2013

Modification	Description	Date de modification
Caractères génériques WLM	La gestion de la charge de travail (WLM) prend en charge les caractères génériques pour l'ajout de groupes d'utilisateurs et de groupes de requêtes aux files d'attente. Pour de plus amples informations, veuillez consulter Caractères génériques.	1er août 2013
Délai WLM	Pour limiter le délai imparti aux requêtes dans une file d'attente WLM donnée, vous pouvez définir la valeur du délai WLM pour chaque file d'attente. Pour de plus amples informations, veuillez consulter Délai WLM.	1er août 2013
Nouvelles options de la commande COPY, « auto » et « epochsecs »	La commande COPY effectue une reconnaissance automatique des formats de date et d'heure. Les nouveaux formats d'heure, « epochsecs » et « epochmillisecs » permettent à la commande COPY de charger les données au format epoch.	25 juillet 2013
Fonction CONVERT_TIMEZONE	La fonction Fonction CONVERT_TIMEZONE convertit un horodatage d'un fuseau horaire à un autre.	25 juillet 2013
Fonction FUNC_SHA1	Fonction FUNC_ SHA1 Convertit une chaîne à l'aide de l' SHA1 algorithme.	15 juillet 2013
max_execution_time	Pour limiter le délai imparti aux requêtes, vous pouvez définir le paramètre max_execution_time dans le cadre de la configuration de WLM. Pour de plus amples informations, veuillez consulter Modifier la configuration WLM.	22 juillet 2013
Caractères UTF-8 à quatre octets	Le type de données VARCHAR prend désormais en charge les caractères UTF-8 à quatre octets. Les caractères UTF-8 à cinq octets et plus ne sont pas pris en charge. Pour de plus amples informations, veuillez consulter Stockage et plages.	18 juillet 2013

Modification	Description	Date de modification
SVL_QERROR	La vue système SVL_QERROR est obsolète.	12 juillet 2013
Version du document révisée	La page Historique du document présente maintenant la date de mise à jour de la documentation.	12 juillet 2013
STL_UNLOAD_LOG	La fonction STL_UNLOAD_LOG enregistre les détails d'une opération de déchargement.	5 juillet 2013
Paramètre de taille d'extraction JDBC	Pour éviter les erreurs de saturation de mémoire côté client lors de l'extraction d'ensembles de données volumineux à l'aide de JDBC, vous pouvez autoriser votre client à extraire les données par lots en définissant le paramètre de taille d'extraction JDBC. Pour de plus amples informations, veuillez consulter Définition du paramètre de taille d'extraction JDBC.	27 juin 2013
Application de la fonction UNLOAD sur des fichiers chiffrés	UNLOAD prend désormais en charge le déchargement des données de table vers des fichiers chiffrés sur Amazon S3.	22 mai 2013
Informations d'identification temporaires	Les fonctions COPY et UNLOAD prennent désormais en charge l'utilisation d'informations d'identification temporaires.	11 avril 2013
Ajout de précisions	Discussions précisées et étendues sur la conception des tables et le chargement des données.	14 février 2013
Ajout de bonnes pratiques	Ajout de Bonnes pratiques Amazon Redshift pour la conception de tables et de Bonnes pratiques de chargement des données sur Amazon Redshift.	14 février 2013
Clarification des contraintes relatives à un mot de passe	Clarification des contraintes relatives à un mot de passe pour CREATE USER et ALTER USER, diverses révisions mineures.	14 février 2013

Modification	Description	Date de modification
Nouveau guide	Ce document est la première version du Guide du développeur Amazon Redshift.	14 février 2013

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.